FACULTES UNIVERSITAIRES NOTRE-DAME DE LA PAIX
Institut d'Informatique

Les premiers pas d'un intranet à l'Institut :

Etude conceptuelle et réalisation d'un prototype d'application

Frédéric Filée

Mémoire présenté en vue de l'obtention du titre de Licencié en Informatique, sous la direction de Monsieur Roland Lesuisse

Année académique 1998-1999

Remerciements

Ce travail n'aurait pu aboutir sans le soutien et les conseils de mon Promoteur, Monsieur Roland Lesuisse, que je remercie chaleureusement.

Je suis également redevable à bien des membres de l'Institut. Ainsi, les avis techniques de Messieurs Radu Cotet, Jean Henrard et Vincent Letocart m'ont apporté une aide précieuse. De même, Mesdames Gysèle Henrard, Anne-Marie Brény et Babette di Guardia n'ont pas hésité à me donner de leur temps.

J'adresse aussi mon entière reconnaissance à toutes les personnes de mon entourage qui m'ont accompagné et encouragé durant la rédaction de ces pages. Je pense à toute ma famille, et particulièrement à mes parents, à Madame Marie-Jeanne Gosselin-Vincart, à Madame et Monsieur Marie-Rose et Jean-Pierre Gosselin et plus largement à toute leur famille, à Mademoisselle Brigitte Malherbe, à Monsieur Koen Anrijs, à Monsieur Christophe Rousseau et à tous mes condisciples de deuxième Licence informatique.

Citer chacun est impossible. Que tous ceux que j'ai omis puissent trouver dans ces lignes toute l'expression de ma gratitude.

Enfin, je tiens à rendre hommage à celle qui a partagé, jour après jour, mes peines et mes joies. C'est grâce à elle que je suis passé de la philosophie à l'informatique. Et c'est grâce à elle que j'ai trouvé le courage d'achever cette Licence. A toi, et au Petit Ange que tu portes, je dédie ce travail.

Sommaire

TABLE DES FIGURES 5

INTRODUCTION GÉNÉRALE 7

CHAPITRE I. LES INTRANETS : PRINCIPES DE BASE 11

I.1. Les intranets : Internet ou Net interne ? 11
I.1.1. Le lien à l'Internet 11
I.1.2. Un outil de communication privé 12
I.1.3. Un prestataire de services 13
I.1.4. Définition 14

I.2. L'infrastructure requise 14
I.2.1. L'infrastructure matérielle 15
I.2.2. L'infrastructure logicielle 16
I.2.3. Les fondements protocolaires 17

I.3. Méthodes techniques de construction d'un intranet 20
I.3.1. Le poste client 20
I.3.2. Le serveur 21
I.3.3. La sécurité de l'intranet 23

I.4. L'architecture intranet 24
I.4.1. L'architecture tripartite 24
I.4.2. Le middleware 25
I.4.3. Le moniteur transactionnel 26

I.5. Les services rendus par un intranet 27
I.5.1. La distribution d'informations 27
I.5.2. La distribution d'applications 28

I.6. Les impacts organisationnels potentiels 29
I.6.1. L'efficacité 29
I.6.2. La qualité 30
I.6.3. La flexibilité 30
I.6.3.1. La situation organisationnelle 31
I.6.3.2. La solution technologique 32

Conclusion du chapitre I 34

CHAPITRE II. UN INTRANET À L'INSTITUT 37

II.1. La mise en place d'un intranet : éléments de théorie........ 37
II.1.1. Quelques principes généraux 37
II.1.2. Le projet intranet........ 38
II.1.2.1. La préparation 38
II.1.2.2. Le développement incrémental 40
II.1.2.3. L'évaluation continue 40

II.2. L'existant à l'Institut 41
II.2.1. Description........ 41
II.2.2. Analyse 42
II.2.2.1. Les pré-requis de l'intranet 42
II.2.2.2. Aspects politiques, techniques et fonctionnels........ 43
II.2.3. Prolongements 44

II.3. Un relevé des besoins 45
II.3.1. Les besoins administratifs........ 45
II.3.1.1. Le trombinoscope........ 45
II.3.1.2. Le serveur de fax........ 46
II.3.1.3. L'agenda partagé........ 46
II.3.1.4. Le service « emploi » 47
II.3.1.5. Les inscriptions (examens, cours à option, autres sections) 48
II.3.1.6. Une base de données des membres de l'Institut 49
II.3.1.7. Les comptes-rendus des réunions et des prises de décision 50
II.3.1.8. La gestion des indisponibilités des professeurs........ 50
II.3.1.9. Les commandes à l'économat........ 51
II.3.1.10. La liste des étudiants pour les entreprises........ 51
II.3.1.11. Les tractations avec les agences de voyage........ 52
II.3.2. Les besoins pédagogiques........ 53
II.3.2.1. L'horaire des cours en temps réel 53
II.3.2.2. La publication du programme analytique des cours 54
II.3.2.3. La publication du guide de l'étudiant........ 54
II.3.2.4. Les propositions et choix de mémoire 55
II.3.2.5. Les valves électroniques 55
II.3.3. Les besoins scientifiques 57
II.3.3.1. Le « welcome staff pack » électronique........ 57
II.3.3.2. Une liste des ouvrages et des mémoires disponibles........ 57
II.3.3.3. La publication des bourses possibles 58
II.3.3.4. L'organisation de colloques 58

II.4. Essai d'évaluation (*scoring*)........ 59
II.4.1. Choix des critères d'évaluation 59
II.4.2. *Scoring* des besoins........ 61
II.4.3. Structuration des besoins 64
II.4.3.1. Les projets abandonnés 65
II.4.3.2. Les projets sous conditions........ 66
II.4.3.3. Les projets à approfondir 66
II.4.3.4. Les publications faciles 66
II.4.3.5. Les priorités........ 67

Conclusion du chapitre II........ 67

CHAPITRE III. LES VALVES ÉLECTRONIQUES 69

III.1. Les valves électroniques : quelques précisions........ 69
III.1.1. Précision des besoins........ 69
III.1.1.1. La situation actuelle........ 70
III.1.1.2. La migration électronique........ 70
III.1.2. Les contraintes techniques et les choix de développement........ 72
III.1.2.1. L'application des principes généraux........ 72
III.1.2.2. La rapidité de consultation........ 72
III.1.2.3. La sécurité des valves........ 74
III.1.3. Contraintes organisationnelles........ 74
III.1.3.1. La charge de travail........ 74
III.1.3.2. La distribution de l'information et des pouvoirs........ 74
III.1.4. Synthèse........ 75

III.2. Le schéma entité-association........ 76

III.3. L'architecture de l'application........ 80
III.3.1. Spécifications des objets de l'application........ 80
III.3.2. Une architecture orientée objet........ 82
III.3.2.1. Les méthodes de l'objet « avis »........ 83
III.3.2.2. Les méthodes de l'objet « valve »........ 83
III.3.2.3. L'objet erreur........ 84
III.3.2.4. Représentation graphique de l'architecture objet........ 84
III.3.3. Une architecture centrée sur les traitements........ 85
III.3.3.1. L'ajout d'un avis........ 85
III.3.3.2. La suppression d'un avis........ 89
III.3.3.3. La mise à jour des valves........ 93
III.3.4. La répartition des modules dans les scripts........ 96

III.4. L'interface homme-machine........ 98
III.4.1. La consultation des valves........ 98
III.4.2. Le formulaire d'ajout d'avis aux valves........ 100
III.4.3. Le formulaire de suppression d'avis aux valves........ 101

III.5. Les possibilités d'évolution........ 103
III.5.1. Changer le *Webmaster* des valves........ 103
III.5.2. Diminuer le temps de réponse........ 104
III.5.3. Mieux gérer l'indisponibilité des valves........ 104
III.5.4. Ajouter une nouvelle valve........ 105
III.5.5. Différencier l'importance des avis........ 106

Conclusion du chapitre III........ 108

CONCLUSION GÉNÉRALE........ 111

BIBLIOGRAPHIE........ 115

ANNEXE 1. SPÉCIFICATIONS DES MODULES DE L'APPLICATION ....Annexes page 1

ANNEXE 2. CODE DES SCRIPTS ....Annexes page 17

Annexe 2.1. Le script AjoutAvisValve ....Annexes page 17

Annexe 2.2. Le script ConsultSupprValve..Annexes page 31

Annexe 2.3. Le script SuppressionAvis ....Annexes page 39

Annexe 2.4. Le script MiseAJourValve ....Annexes page 50

ANNEXE 3. CODE DU FORMULAIRE D'AJOUT D'UN AVIS.....................................Annexes page 67

ANNEXE 4. MANUEL D'INSTALLATION DES VALVES...Annexes page 73

Table des figures

Figure 1 : Les couches du modèle OSI et les principaux protocoles 18
Figure 2 : Architecture tri-partite intranet 24
Figure 3 : Tableau de comparaison de l'intranet et des systèmes traditionnels 30
Figure 4 : Les modèles organisationnels 32
Figure 5 : Les critères d'évaluation du scoring.. 60
Figure 7 : Tableau synthétique du scoring des besoins 64
Figure 8 : Graphique de comparaison des apports et des risques.. 65
Figure 9 : Typologie des besoins.. 68
Figure 10 : Tableau des valves à prévoir.. 76
Figure 11 : Le schéma entité-association des valves de l'Institut. 79
Figure 12 : Tableau des marques distinctives d'une valve et d'un avis 81
Figure 13 : Exemple type d'un fichier de valve non vide 81
Figure 14 : Exemple type d'un fichier de valve vide 82
Figure 15 : Architecture orientée objet.. 84
Figure 16 : Les modules généraux de l'architecture top-down 85
Figure 17 : Architecture du module général AjoutAvisValve 87
Figure 18 : Tableau des constantes et des variables globales de AjoutAvisValve................... 88
Figure 19 : Les deux sous-modules du module général SuppressionAvisValve 89
Figure 20 : Architecture du sous-module général ConsultSupprValve.................................... 90
Figure 21 : Tableau des constantes et des variables globales de ConsultSupprValve 91
Figure 22 : Architecture du sous-module général SuppressionAvis 92
Figure 23 : Tableau des constantes et des variables globales de SuppressionAvis.................. 93
Figure 24 : Architecture du module général MiseAJourValve 95
Figure 25 : Tableau des constantes et des variables globales de MiseAJourValve 96
Figure 26 : Tableau de répartition des modules dans les scripts 97
Figure 27 : Copie d'écran d'une valve 99
Figure 28 : Copie d'écran du formulaire d'ajout d'un avis aux valves................................. 101
Figure 29 : Copie d'écran du formulaire de suppression d'un avis dans une valve............... 102

Page blanche

Introduction générale

L'information, dans toutes les organisations actuelles, est capitale : « Information is the way organizations coordinate their activities and achieve their goals »[1]. Elle est la clef de la coordination, de la gestion et du contrôle des activités : « A requirement for successful coordination is consistency of information »[2]. Jusqu'il y a peu, la transmission de cette information reposait sur des systèmes administratifs manuels, avec de nombreux documents en papier, lourds, complexes, lents. Avec l'émergence des nouvelles technologies, et particulièrement celles de la téléinformatique et des réseaux, un nouveau type de communication a vu le jour : la communication électronique.

Celle-ci, au gré des innovations techniques, a conquis progressivement le monde socio-économique, pour s'imposer aujourd'hui comme l'outil par excellence de l'information.

Ainsi des réseaux locaux (*Local Area Network*) ont d'abord été développés. Ces systèmes centralisés ont donné aux organisations les premiers outils de communication électronique : des terminaux, connectés à des *mainframes*, permettaient aux utilisateurs d'échanger des données. Dans un deuxième temps, les solutions client/serveur ont alors pris le relais : « Le début des années quatre-vingt-dix voit se déplacer le cœur de l'entreprise (du site central vers l'utilisateur) par l'introduction de la technologie du client/serveur, c'est-à-dire la recherche d'optimisation de la répartition des données, traitements et communications entre les différentes niveaux de l'architecture informatique de l'entreprise (sites centraux, serveurs départementaux, postes clients connectés) »[3]. Alors que les *mainframes* centralisent le fonctionnement organisationnel, le client/serveur, pour sa part, défère une partie de la charge du réseau sur les postes clients : les informations et les responsabilités qui en découlent connaissent une première décentralisation.

Cependant, la technique du client/serveur privilégie trop les parties sur le tout : elle risque d'atomiser la communication et de briser une distribution uniforme de l'information ; « it [client-server] did not solve the fundamental problem of distributing information in organizations »[4]. De plus la complexité et l'hétérogénéité de cette solution causent de nombreuses difficultés concrètes[5]. Pour remédier à celles-ci, certains constructeurs de logiciels ont élaboré des outils de *groupware*, totalement consacrés à la communication.

---

1 TELLEEN St. L., *The IntraNet Architecture: Managing information in the new paradigm*, Copyright © 1996 Amdahl Corporation, Sunnyvale, California, USA, cf. http://www.amdahl.com/doc/products/bsg/intra/infra.htm.

2 *Ibid.*

3 ALIN Fr., LAFONT D., MACARY J.-Fr., *Le projet intranet. De l'analyse des besoins de l'entreprise à la mise en œuvre de solutions*, Paris, Eyrolles, Coll. Fi System, 1998, p. 38.

4 TELLEEN St. L., *Intranets and Adaptive Innovation: The move from control to coordination in today's organizations*, Copyright © 1996 Amdahl Corporation, Sunnyvale, California, USA, cf. http://www.amdahl.com/doc/products/bsg/intra/adapt.htm.

5 Pour une description plus détaillée des désavantages du client/serveur, cf. ALIN Fr., LAFONT D., MACARY J.-Fr., *op. cit.*, pp. 38-39. Ces auteurs détectent, globalement, les difficultés suivantes au sein de la solution client/serveur : une complication considérable de l'administration du système d'information, le manque de modélisation d'architecture client/serveur, l'absence de systèmes d'aide à la décision, une offre trop variée et trop propriétaire, le syndrome du client obèse, le tâtonnement des outils de deuxième génération à trois niveaux, l'exigence d'environnements compatibles et de plates-formes de développement et d'exploitation universelles.

Malheureusement, ces produits pêchent par leurs avantages mêmes : leur efficacité provient de leur développement propriétaire, qui les limite dans leurs possibilités d'évolution. Les outils de *groupware* sont chers à la construction, coûteux à l'implantation et à la maintenance, et ils exigent une fidélité au constructeur pour assurer une totale compatibilité[6].

L'Internet a alors proposé une solution à une situation qu'il aggrave lui-même. Le paradoxe de son développement se pose en ces termes : d'une part, il engendre des flux toujours plus importants d'informations, qui submergent les individus et les institutions ; et d'autre part, il pose les bases d'une technologie de communication nouvelle : la technologie intranet.

Face à la massification d'informations engendrée par le Web, il devenait urgent d'inventer « new options for more effective coordination of organizational activities in a distributed information environment »[7]. Historiquement, l'intranet se veut ainsi le fruit d'une combinaison des techniques et des concepts des réseaux : après l'emploi des *mainframes*, après l'architecture client/serveur, l'intranet veut réaliser un compromis subtil entre une gestion centralisée de l'information, une optimisation distribuée des systèmes informatiques et une attention particulière aux besoins individuels des utilisateurs : « L'apparition de nombreux projets de migration d'architectures client/serveur vers des systèmes d'information intranet tend à montrer tout l'intérêt que portent les responsables informatiques à ces technologies »[8].

Cet intérêt pour les intranets ne pouvait laisser indifférente une organisation comme l'Institut d'Informatique des Facultés Universitaires Notre-Dame de la Paix. Cette institution académique, comme toute autre organisation contemporaine, est confrontée à la problématique de la distribution de l'information et s'intéresse, elle aussi, aux solutions nouvelles dans ce domaine. L'Institut pressent qu'un intranet pourra lui apporter un système de communication plus performant. Mais, jusqu'ici, seule une étude réalisée par Pascal Goossens a envisagé diverses possibilités liées à la « nouvelle utilisation des réseaux », sans explicitement les inscrire dans un projet intranet[9].

Dans ce mémoire, nous allons tenter d'entreprendre une étude de la solution intranet adaptée à l'Institut. Nous allons nous pencher sur les besoins en termes de communication électronique, afin de mesurer l'utilité d'un intranet pour l'Institut et d'en commencer le développement

Notre démarche doit commencer par une interrogation théorique des principes de base de la technologie intranet : en quoi consiste-t-elle, comment fonctionne-t-elle, que propose-t-elle, où conduit-elle ? Autant de questions théoriques que notre premier chapitre abordera et auxquelles il tentera d'apporter des réponses claires.

---

6 Pour une comparaison précise entre les intranets et les outils de *groupware*, cf. CASSELBERRY R., et al., *Running a Perfect Intranet,* Copyright © 1996 by Que® Corporation, HTML conversion by M/s. LeafWriters (India) Pvt. Ltd., cf. http://www.mcp.com/849139200/0-7897/0-7897-0823-X/index.htm, chapitre 18 : *Groupware Applications*.

7 TELLEEN St. L., *The IntraNet Architecture: Managing information in the new paradigm, art. cit.*

8 ALIN Fr., LAFONT D., MACARY J.-Fr., *op. cit.*, p. 39.

9 Cf. GOOSSENS P., *Nouvelles Utilisations Réseaux*, Namur, F.U.N.D.P. - Institut d'Informatique, document interne.

Dans notre deuxième chapitre, nous pourrons alors nous concentrer sur l'Institut d'Informatique lui-même. Nous étudierons les aspects indispensables à la réalisation d'un intranet afin de vérifier s'ils sont présents dans notre institution académique, nous procéderons à une étude de l'existant afin de cerner les contraintes et les besoins en termes de communication électroniques, puis nous évaluerons et nous classerons les besoins identifiés grâce aux critères que nous nous donnerons. Nous obtiendrons, dès lors, une liste des applications et des informations à déposer sur l'intranet, qui pourrait former une sorte de cahier des charges pour les développements concrets.

Puisqu'il nous est impossible de développer la totalité de ces besoins dans le cadre de ce travail, nous poserons la première pierre de l'intranet de l'Institut en développant l'application considérée comme prioritaire : les valves électroniques. Notre troisième chapitre détaillera la conception et le développement de cette application, tandis que nos annexes présenteront le code des scripts Unix réalisés, ainsi que le code JavaScript du formulaire d'ajout d'avis aux valves.

Page blanche

Chapitre I. Les intranets : principes de base

Le langage de l'informatique paraît souvent hermétique aux non-initiés. Même les professionnels éprouvent des difficultés devant le flot incessant des nouveaux outils et des dénominations originales qui les accompagnent. Les concepts semblent s'enrichir au coup par coup, selon l'usage.

Dans ce premier chapitre, nous nous proposons d'interrompre cette logique inchoative pour considérer l'état actuel des termes qui vont nous intéresser. Nous ne chercherons pas, ici, à donner de définitions définitives, mais plus modestement à éclairer, temporairement, les zones d'ombre.

I.1. Les intranets : Internet ou Net interne ?

Aujourd'hui, le mot « intranet » doit figurer dans le vocabulaire de toute personne s'intéressant de près ou de loin à l'informatique et à l'Internet. S'agit-il d'un pur phénomène de mode, fugace et éphémère, d'une révolution incontournable dans le monde de la communication électronique ou du simple prolongement des développements télé-informatiques ? Pour y voir plus clair, tentons tout d'abord de définir la notion d'intranet.

I.1.1. Le lien à l'Internet

La plupart des informaticiens s'accordent pour définir les intranets eu égard à l'Internet. Ainsi, tous pourraient considérer « un réseau intranet comme l'utilisation au sein de l'entreprise des technologies de l'Internet »[10]. L'intranet, en quelque sorte, est un « site Web interne d'entreprise »[11], et plus généralement de toute organisation quelle qu'elle soit. La première définition d'un intranet que nous pouvons fournir serait donc un *Organization Wide Web*, un site Web interne d'organisation.

Avant toute chose, il importe de clarifier le terme « internet ». Strictement, « internet » désigne une des couches du modèle de référence TCP/IP (*Transmission Control Protocol* / *Internet Protocol*) dans la théorie des réseaux. Elle correspond précisément à la première couche de ce modèle, qui en compte quatre (la couche internet, la couche transport, la couche application et la couche hôte-réseau). Tanenbaum définit la couche internet dans le modèle TCP/IP comme « une couche d'interconnexion de réseaux sans connexion. […] Son rôle est de permettre l'injection de paquets dans n'importe quel réseau et l'acheminement de ces paquets indépendamment les uns des autres jusqu'à destination. […] On peut dire que la couche internet du modèle TCP/IP a des fonctionnalités semblables à celles de la couche

---

10 ALIN Fr., LAFONT D., MACARY J.-Fr., *Le projet intranet. De l'analyse des besoins de l'entreprise à la mise en œuvre de solutions*, Paris, Eyrolles, Coll. Fi System, 1998, p. 13.

11 GAUTIER R.., *Qu'est-ce qu'un intranet ?*, Communication Jean Lalonde 1999, cf. *www.cjl.qc.ca/batisseurs/intranet.htm*

réseau du modèle OSI »[12]. Mais le mot « internet », pour la plupart de nos contemporains, n'évoque pas d'abord ce service de routage inhérent à toute interconnexion d'ordinateurs. Plus communément, l'Internet, que nous orthographierons dans cette signification avec une majuscule, désigne le *World Wide Web*, ou encore la « Toile », c'est-à-dire « Le » réseau de réseaux à portée mondiale : « La Toile (*World Wide Web*) est un concept permettant l'accès à des documents reliés entre eux et disséminés sur des milliers de machines réparties dans le monde entier »[13]. Définir précisément l'Internet exigerait une œuvre aussi colossale que celle de Tanenbaum : nous préférons y renvoyer les lecteurs intéressés et nous concentrer sur les intranets.

Cette référence au réseau internet pour définir un intranet apparaît d'emblée comme fondamentale. Et pourtant, un article quelque peu polémique dénonce l'affirmation selon laquelle « Intranets are internal webs »[14] comme un des mythes entourant les intranets. Selon l'auteur anonyme, les intranets ne se limitent pas à des sites Web, et le croire reviendrait à commettre une grave erreur. Les intranets, en effet, offrent des possibilités très nombreuses, qui en font des entités bien plus complexes et bien plus puissantes que de simples pages déposées sur le réseau des réseaux.

Il faut dès lors nous interroger sur le lien unissant l'intranet à l'Internet : quelles similitudes et quelles différences existe-t-il entre ces deux notions ? Cherche-t-on à reproduire un macrocosme cybernétique dans un microcosme organisationnel ? Pourquoi avoir besoin d'un Internet privé ?

I.1.2. Un outil de communication privé

La réponse à ces questions tient dans l'essence même de la Toile : la communication d'informations. Si l'intranet se réfère à l'Internet, c'est parce qu'il est, lui aussi, orienté vers la communication. Puisque fondamentalement le but est le même, l'intranet peut utiliser la même base technologique : « An Intranet is a communication infrastructure. It is based on the communication standards of the Internet and the content standards of the World-Wide Web »[15].

Cependant, un intranet n'est pas l'Internet. Alors que le Web est mondial, ouvert à tous, l'intranet, pour sa part, s'adresse à une communauté restreinte d'utilisateurs. Ces derniers sont précisément les membres de l'organisation, dans laquelle l'intranet assume la fonction de moyen de communication : « Intranet désigne l'utilisation des technologies d'Internet au sein d'une entreprise par opposition à une utilisation tournée vers l'extérieur, à

---

[12] TANENBAUM Andrew, *Réseaux*, Paris/London, Inter-Editions/Prentice-Hall, 1997, 3ème édition, p. 32. Ce modèle de référence TCP/IP ne doit pas être confondu avec le modèle de référence OSI (*Open Systems Interconnection*), qui dénombre sept couches dans un réseau informatique (la couche physique, la couche liaison de données, la couche réseau, la couche transport, la couche session, la couche présentation et la couche application).

[13] *Ibid.*, p. 684.

[14] *10 Intranet Myths*, Smart Infotech Systems Ltd., Copyright © 1997 by Smart Infotech Systems Ltd., cf. www.intrack.com/intranet/intmyth10.html

[15] TELLEEN St. L., *The IntraNet Architecture: Managing information in the new paradigm*, Copyright © 1996 Amdahl Corporation, Sunnyvale, California, USA, cf. http://www.amdahl.com/doc/products/bsg/intra/infra.htm

savoir le réseau mondial Internet »[16]. Intranet et Internet forment, en quelque sorte, les deux visages de la communication électronique, l'un est privé, l'autre public ; « The World-Wide Web, and its behind-the-firewall couterpart, the Intranet »[17].

Mais un intranet ne peut se réduire à une collection d'informations à caractère privé. Plus profondément, un intranet répond au désir de structurer, de comprendre et de transmettre l'immense masse des informations qui traversent une structure organisationnelle : « Il existe beaucoup de visions différentes ou complémentaires d'un intranet. En se restreignant aux aspects technologiques, nous définissons un intranet comme étant l'utilisation de tout ou partie des technologies et des infrastructures de l'Internet pour transporter et traiter les flux d'informations internes d'un groupe d'utilisateurs identifiés »[18].

Un intranet permet donc, au sein de l'organisation concernée, « de distribuer de manière transparente les informations d'une entreprise sur chacun de ses ordinateurs personnels avec un minimum de coûts, d'efforts et de délais »[19]. Les personnes ainsi reliées entre elles peuvent améliorer leurs communications, leur collaboration et la gestion de leur travail. Elles peuvent, éventuellement, utiliser ou distribuer des applications interactives via un interface unique, celui du navigateur Web.

I.1.3. Un prestataire de services

Un intranet ne se limite donc pas à une structure significative de pages HTML remplies de renseignements. Il peut surtout offrir des services de communication et de gestion, auxquels les personnes peuvent avoir facilement accès. Selon Microsoft, « un Intranet est une combinaison des technologies réseau (LAN et WAN) et Internet, qui permet facilement à n'importe quel utilisateur dans l'entreprise, où qu'il soit, d'accéder aux personnes, aux applications et aux informations dont il a besoin pour travailler efficacement »[20].

Bien au-delà d'un nouvel outil de communication, un Internet privé représente souvent un effort d'intégration des divers composants d'une organisation au sein d'une structure générale commune. « Le site Web interne sert à relier des systèmes parfois incompatibles entre différents services et fonctions de l'entreprise. Grâce à une interface commune multifonctionnelle, un navigateur Web en fait, l'*intranet* permet d'offrir de l'information sur demande grâce à la mise en commun des bibliothèques numériques de données (textes, tableaux, sons, images fixes et animées) produites par l'ensemble du personnel. Les données circulent sur le réseau, conformément aux responsabilités et codes d'accès de chacun, elles sont à jour et accessibles à l'usager qui peut alors les utiliser telles quelles ou les reformater selon ses besoins. Bref, l'*intranet* sert à travailler en groupe, à partager documents et applications, pour communiquer plus efficacement. L'entreprise dispose alors d'un espace ouvert de ressources à portée de souris de chacun. De là l'expression "solutions collecticielles", c'est-à-dire l'émergence d'outils avec lesquels il est possible de travailler de

---

[16] www.radian.fr/intranet.html, Radian Communication 1999.

[17] TELLEEN St. L., *Intranets and Adaptive Innovation: The move from control to coordination in today's organizations*, Copyright © 1996 Amdahl Corporation, Sunnyvale, California, USA, cf. http://www.amdahl.com/doc/products/bsg/intra/adapt.htm.

[18] ALIN Fr., LAFONT D., MACARY J.-Fr., *op. cit.*, p. 45.

[19] Cf. www.radian.fr/intranet.html, Radian Communication 1999.

[20] Cf. www.microsoft.com/france/intranet/.

façon intégrée, en collectivité, plutôt qu'en vase clos. Le partage des connaissances serait-il l'objet véritable de la révolution de l'information tant promise depuis des lunes? »[21]. Il s'agit donc d'offrir, d'une manière simple, efficace et distribuée, les ressources applicatives et communicationnelles à tous les membres de l'organisation.

Vraisemblablement, les intranets signent l'avènement de nouveaux systèmes d'information : « These new enterprise information systems are called Intranets, and represent the beginning of a new computing paradigm »[22]. En effet, les intranets ne sont pas des outils comme les autres, qu'il suffirait purement et simplement d'intégrer dans le système d'information officiel de l'entreprise ; les informations qu'un intranet permet de véhiculer concernent également toutes les « coulisses » de l'organisation, c'est-à-dire tous les traitements officieux qu'un S.I. traditionnel ne peut cerner [23]. Dans une organisation, l'intranet peut ainsi constituer un complément au S.I. afin d'assurer une plus grande fluidité dans la distribution de l'information.

I.1.4. Définition

Pour conclure avec cette tentative de définition, nous pouvons rassembler les caractéristiques relevées en une seule proposition :

> « un intranet constitue un outil de communication, basé sur les technologies de l'Internet, consacré à une organisation et à ses membres, pour offrir à ces derniers les ressources informationnelles, matérielles et logicielles de l'organisation, afin qu'ils les exploitent et les enrichissent, avec plus d'efficacité, de souplesse, de dynamisme et de facilité ».

I.2. L'infrastructure requise

Après avoir établi une définition abstraite du concept d'intranet, nous pouvons nous pencher sur les aspects technologiques qu'il implique. Ces derniers sont nombreux et diversifiés. Toute la théorie des réseaux est ici concernée. Dans le cadre de ce travail, nous ne pouvons l'aborder systématiquement et exhaustivement. L'ouvrage de Tanenbaum, que nous avons déjà cité, constitue une source précieuse, à laquelle nous renvoyons[24]. De plus, les développements actuels débouchent sans cesse sur la mise au point de nouveaux outils, ou du moins sur des versions toujours améliorées. Dans cette section, nous nous limiterons à la présentation des techniques des intranets. Cette vision de base devrait nous permettre de comprendre les principes théoriques, sans nous perdre dans le dédale des nuances.

---

21 GAUTIER R., *art. cit.*

22 *The Intranet. Implementation of Internet And Web Technologies*, Organizational Information Systems, Hummingbird Communications Ltd., 1996, cf. www.hummingbird.com/whites/intranet.html.

23 Cf. *10 Intranet Myths*, Smart Infotech Systems Ltd., Copyright © 1997 by Smart Infotech Systems Ltd., cf. www.intrack.com/intranet/intmyth10.html.

24 Cf. TANENBAUM Andrew, *op. cit.*.

I.2.1. L'infrastructure matérielle

La condition matérielle indispensable d'un intranet est l'existence d'un réseau physique local ou distant. Le réseau local peut prendre, par exemple, la forme d'un anneau à jeton, d'un bus à jeton, d'un Ethernet ou d'un ATM ; comme réseau distant, on peut trouver du X25, du Frame Relay, du PPP ou du RNIS[25]. Pour supporter un intranet, il faut « a minimum of a 100 Mpbs [megabit-per-second] network with T-1 (1.544 Mbps) and T-3 (45 Mbps speed) lines to the Internet »[26]. La question, souvent épineuse, de la bande passante doit être abordée en fonction des buts assignés à l'intranet : il est évident que la vidéo-conférence nécessite plus de ressources que le courrier électronique. Comme toujours en informatique et dans les sciences de la communication, il faut trouver l'équilibre entre le coût des technologies et les besoins de l'organisation : dépasser les besoins coûtera souvent cher et ne permettra pas de retour sur l'investissement ; ne pas respecter les besoins exprimés dans le cahier des charges signifiera vraisemblablement l'inutilité du résultat, et donc son échec.

Le réseau physique ne suffit pas à un intranet. Il faut également un « serveur ». Les serveurs ne sont pas nécessairement des produits *hardware* ; ce sont souvent des solutions logicielles, mais tellement spécifiques qu'elles doivent de préférence tourner sur une machine qui leur est dédiée. Ce lien entre le matériel et le logiciel justifie que nous présentions les serveurs dans cette section.

De nombreux serveurs existent sur le marché, depuis les systèmes NT de Windows, jusqu'aux stations Unix, en passant par les multiples solutions propriétaires. Sun a notamment proposé un serveur, dans la gamme des « SPARCstations », spécifiquement orienté vers les intranets : le « Netra ». Les Netras i 4000 et i 5000 sont ainsi totalement prévus pour une utilisation dans le cadre de l'internet et des intranets, tant au point de vue logiciel que matériel[27].

Nous pouvons ainsi mentionner le serveur Apache, orienté Unix, développé par le « *National Center for Supercomputing at the University of Illinois at Champaign-Urbana* », et proposé gratuitement au public. Microsoft, Amiga, Netscape, Novell et bien d'autres ont également mis au point des solutions hardware et software qui tiennent lieu de serveur. Le choix entre ces serveurs est bien souvent difficile, car les solutions ne sont pas compatibles. Prendre tel ou tel serveur, c'est souvent s'attacher au constructeur et à la gamme des produits qu'il propose : un compromis entre finances et besoins doit de nouveau être trouvé.

Avec un réseau physique et un serveur, les conditions matérielles pour l'existence d'un intranet sont réunies. C'est nécessaire, mais pas suffisant.

---

25 Pour en savoir plus sur les caractéristiques physiques des réseaux, cf. TANENBAUM Andrew, *op. cit.*

26 ESPLIN K., *8 important issues to consider before building an intranet, What must you know about your intranet's infrastructure and staffing to be successful? These guidelines will help you through the morass of intranet planning. Plus Sun and Lockheed Martin tell how they constructed their massive intranets,* Mars 1997, cf. www.sunworld.com/swol-03-1997/swol-03-intranet.html.

27 Ces serveurs possèdent une interface HTML, supportent un outil Netscape de navigation et de communication, contiennent des programmes de gestion et de sécurisation des réseaux et connaissent le protocole FTP. Ils supportent également le langage Java, et ouvrent ainsi la voie à des développements à la fois plus pointus et plus homogènes.

I.2.2. L'infrastructure logicielle

Pour cerner les besoins logiciels d'un intranet, il faut distinguer les serveurs des postes clients. Les postes clients du réseau doivent posséder un programme de consultation des pages Web. Ce programme de consultation porte le nom de *browser*, traduit par « navigateur » ou par « fureteur »[28]. Les principaux navigateurs actuels sont Internet Explorer de Microsoft et Netscape Navigator and Communicator de Netscape[29]. Le navigateur prend en charge la mise en page de l'information qu'il reçoit du serveur, compte tenu des particularités de l'ordinateur sur lequel il tourne (taille de l'écran, résolution…). Par ailleurs, les navigateurs actuels sont aussi capables d'interpréter et d'exécuter eux-mêmes des programmes, ou au moins de constituer l'interface d'exécution de programmes indépendants : des *plug-ins* viennent continuellement augmenter leurs potentialités.

Le navigateur constitue ainsi le point d'entrée unique du poste client sur le monde de l'Internet et des intranets. Cette concentration des informations et des applications en une seule interface procure une grande simplicité, et dès lors une facilité d'apprentissage importante : c'est l'*Universal Client concept*[30]. Du point de vue de l'utilisateur, l'intranet peut être considéré comme un seul « programme », aux multiples facettes.

Tous les navigateurs actuels se valent. Mais, compte tenu de légères et problématiques nuances existant entre eux, il vaut mieux rester fidèle au choix qu'on a posé. Ainsi, on est sûr que les informations publiées sont reçues de la même manière par tous.

A côté du navigateur, on pourra également trouver, sur le poste client, un programme de messagerie électronique. Ce type de logiciel n'est pas indispensable, comme tel, à un intranet, mais il s'avère très utile pour établir des communications électroniques personnalisées.

Du côté du serveur, des logiciels doivent sélectionner et envoyer des pages Web et éventuellement les créer. Deux types de serveurs Web existent : les serveurs Web statiques, ou serveurs HTTP, qui renvoient les documents demandés sans aucun traitement préalable (le serveur Apache, gratuit dans le monde Unix, par exemple), et les serveurs Web dynamiques, ou serveurs Web applicatifs, qui construisent une page Web en fonction d'une requête précise. Si le premier type de serveur se contente de programmes simples utilisant les mécanismes inhérents au protocole HTTP (que nous présenterons au point suivant), le deuxième requiert des programmes spécifiques souvent écrits sur mesure. Il faut alors véritablement construire l'intranet. Bien d'autres logiciels peuvent encore trouver leur place sur un intranet. Nous ne présenterons dans cette section que ceux qui sont nécessaires. Non seulement, on peut encore en construire, mais on peut également en acquérir. Ainsi nous pouvons citer, entre autres : un éditeur HTML pour publier l'information, une boîte à outils de scripts constituant un CGI, un gestionnaire de base de données (l'accès au SQL peut être opéré à partir d'Oracle, d'Informix, de Sybase, voire même d'applets Java), des applications de développement des pages Webs (Microsoft Front Page, Adobe Page Mill, HoTMetal, Hotdog…).

---

[28] Cf. TANENBAUM Andrew, *op. cit.*, p. 685.

[29] On pourrait également citer NCSA Mosaic (premier navigateur de l'histoire, issu du *National Center for Supercomputer Applications* de l'Université d'Illinois) et Lynx, mais ces deux navigateurs tendent à disparaître.

[30] Cf. TELLEEN St. L., *IntraNet Methodology. Concepts and Rationale*, Copyright © 1996 Amdahl Corporation, Sunnyvale, California, USA, cf. http://www.amdahl.com/doc/products/bsg/intra/concept.htm.

I.2.3. Les fondements protocolaires

Plus profondément que le matériel ou le logiciel, ce sont les protocoles employés qui structurent et caractérisent les intranets. Ainsi, concrètement, la ressemblance fondamentale entre l'Internet et les intranets se traduit par l'emploi commun du protocole TCP/IP (*Transmission Control Protocol / Internet Protocol*) : « TCP/IP est le ciment d'un intranet »[31]. Techniquement, cela recouvre deux protocoles, qui correspondent à deux couches dans la théorie des réseaux[32]. Le protocole IP appartient à la couche réseau du modèle OSI, qui est responsable du routage des données à travers les chemins possibles du réseau et des contrôles de congestion des lignes de communication. TCP, pour sa part, relève de la couche transport. C'est un protocole orienté connexion, qui assure que les données de la couche IP sont arrivées intactes (grâce à un accusé de réception) et dans le bon ordre ; il assure donc une fiabilité à un réseau ou à un ensemble de réseaux qui en est dépourvu, indépendamment de la nature de ce(s) réseau(x). C'est ainsi que les « datagrammes » IP, qui contiennent les données fournies par les couches inférieures (la couche physique et la couche liaison de données), sont eux-mêmes encapsulés dans des « segments » TCP, qui pourront être récupérés par les couches supérieures successives (les couches session, présentation puis application). Le couple TCP/IP peut ainsi assurer la transmission d'informations sur n'importe quel réseau physique, aussi bien en réseau local qu'en réseau distant.

Hormis TCP, l'internet utilise également un protocole transport appelé UDP (*User Data Protocol*) : UDP n'est pas orienté connexion, il n'assure ni l'intégrité des données reçues ni leur ordre d'arrivée, mais il implique moins de traitements des données. Il se contente d'ajouter un petit en-tête au datagramme IP. Il convient bien pour des applications de type « question-réponse », comme le service DNS (*Domain Name Service*) [33], qui convertit les noms symboliques en adresses IP, ou le protocole RIP (*Routing Information Protocol*), qui s'occupe des informations de routage. Les applications dites « temps réel », qui préfèrent recevoir une information endommagée plutôt qu'attendre sa retransmission, fonctionnent également avec UDP. Les deux couples TCP/IP et UDP/IP constituent en quelque sorte l'ossature de l'Internet et des intranets.

---

[31] ALIN Fr., LAFONT D., MACARY J.-Fr., *op. cit.*, p. 142.

[32] Comme nous l'avons déjà souligné, nous ne présenterons que très sommairement les divers protocoles du monde Internet-intranet dans ces pages. Pour des explications à la fois plus techniques et plus précises, cf. TANENBAUM Andrew, *op. cit.* :

- pour IP, cf. pp. 432-442 ;
- pour TCP et UDP, cf. pp. 541-562 ;
- pour IPv6 vs IPv4, cf. pp. 460-471 :
- pour les réseaux NetWare de Novell, cf. pp. 40-42 ;
- pour DNS, cf. pp. 631-639 ;
- pour SMTP, cf. pp. 664-667, dont il existe une version étendue : ESMTP ;
- pour POP3 et IMAP4, cf. p. 667 ;
- pour MIME, cf. pp. 659-663, dont il existe une version plus sécurisée : SMIME ;
- pour HTTP, cf. pp. 692-694 ;
- pour URL et HTML, cf. pp. 694-707 ;
- pour CGI, cf. p. 707 ;
- pour Java, cf. pp. 710-719.

[33] Les auteurs du livre *Le projet intranet, De l'analyse des besoins de l'entreprise à la mise en œuvre de solutions* placent le DNS au sein des protocoles réseaux (cf. *op. cit.*, pp. 143 et 150). Ils commettent une erreur, le DNS appartient à la couche application, comme le confirme Tanenbaum, cf. *op. cit.*, pp. 631 ss.

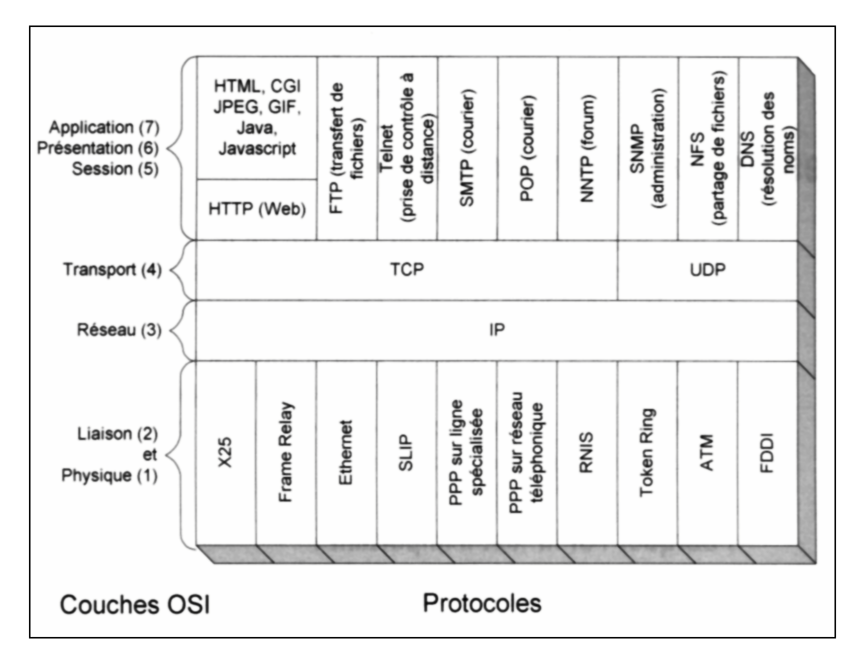


Figure 1 : Les couches du modèle OSI et les principaux protocoles
Issu de ALIN Fr. et al., *op. cit.*, p. 144.

Mais, même dans le standard de l'internet, des nuances doivent être introduites. La version actuelle du protocole IP, qui est la numéro quatre, semble d'ores et déjà désuète. La pénurie d'adresses IP qu'elle entraîne, le manque d'attention accordée au type de service requis et les carences de sécurité de cette version ont entraîné la recherche vers un nouveau protocole : IPv6 (IPv5 étant le nom d'un protocole temps réel expérimental). Pour tous les auteurs, IPv6 s'impose comme le successeur inévitable d'IPv4, mais à long terme : « une décennie »[34], « plusieurs années »[35]. Vraisemblablement, les intranets de demain fonctionneront avec cette sixième mouture du protocole IP.

TCP/IP et UDP/IP se limitent aux couches réseau et transport. Sur ces protocoles, viennent alors se greffer toute une série d'autres protocoles, qui appartiennent à la couche « application » du modèle de référence OSI.

Pour permettre la publication et la consultation de documents *on-line*, les intranets, tout comme l'Internet, utilisent des protocoles d'échanges de pages d'informations. Un couple heureux s'occupe de la transmission et de la réception de ces pages dans le monde Internet/intranet : HTTP (*HyperText Transfer Protocol*) et HTML (*HyperText Markup Language*). HTTP s'occupe du transfert des documents, et HTML de leur présentation.

Le protocole HTML consiste en une série de balises (*tags*), destinées au navigateur, qui permettent de marquer le texte à afficher. Ces balises permettent d'introduire des commandes de formatage du texte, qui seront interprétées et exécutées par le navigateur. Ainsi, il est possible de créer un document avec HTML qui aura globalement la même présentation, quel que soit l'ordinateur et le navigateur qui le recevra.

---

[34] TANENBAUM Andrew, *op. cit.*, p. 471.

[35] ALIN Fr., LAFONT D., MACARY J.-Fr., *op. cit.*, p. 153.

Pour construire des pages HTML, un simple éditeur de texte suffit, à condition de connaître les balises nécessaires. De nombreux outils d'aide à la publication HTML existent aujourd'hui sur le marché, du simple éditeur de code HTML jusqu'aux outils proches des traitements de texte, en passant par des fonctions de conversion diverses. Par convention, toutes les pages HTML portent un nom unique au niveau mondial : c'est leur URL (*Uniform Ressource Locator*). L'URL d'une page révèle à la fois le protocole mis en œuvre, le nom DNS de la machine de la page et le nom local de la page : par exemple, http://www.info.fundp.ac.be/index.html forme une URL valide. Toutes les pages d'un intranet recevront ainsi une URL spécifique.

Pour transférer les pages HTML du serveur (où elles sont stockées ou fabriquées) vers les postes clients, c'est le protocole HTTP qui est employé par le monde de l'Internet. Conformément à une philosophie délibérée d'orientation objet, ce protocole contient plusieurs « méthodes », plusieurs commandes, à exécuter sur des pages Web. La méthode GET appelle une page Web (considérée comme un objet) selon un format spécifié ; la méthode HEAD capture l'en-tête d'une page ; PUT écrit une page ; POST ajoute des données au sein d'une page existante ; DELETE efface la page ; LINK et UNLINK établissent ou suppriment des connexions entres des pages. A l'aide de ces primitives, un serveur peut donc répondre aux requêtes simples des clients. Pour garantir le format des informations véhiculées, on s'appuiera sur le protocole MIME (*Multipurpose Internet Mail Extensions*) qui définit, grâce à des en-têtes spécifiques, les règles du codage pour les messages non ASCII.

A côté de ces deux protocoles fondamentaux de la couche application, il en existe d'autres, davantage orientés vers des applications spécifiques qu'on retrouve souvent dans les intranets.

Ainsi le protocole SMTP (*Simple Mail Transfer Protocol*) gère essentiellement les échanges entre les serveurs de messagerie électronique (comparables à des bureaux de poste). C'est un simple protocole ASCII qui permet d'accepter ou non les connexions entrantes et de copier les messages reçus dans les bonnes boîtes aux lettres[36]. Pour envoyer et recevoir des messages à partir d'un poste client (un ordinateur hôte local du réseau), les protocoles POP3 (*Post Office Protocol*) ou IMAP4 (*Interactive Mail Access Protocol*) peuvent être utilisés. POP3 prend le courrier dans la boîte distante et le rapatrie sur le poste local. IMAP4, pour sa part, grâce à un référentiel central auquel plusieurs ordinateurs peuvent accéder, peut satisfaire l'utilisateur qui dispose de plusieurs ordinateurs ; mais il ne permet pas de copier le courrier dans l'ordinateur client. Le protocole MIME permettra de nouveau d'obtenir une convention de codage.

Dans le cadre des organisations de taille importante, un intranet fonctionnera souvent avec un ou plusieurs annuaires, qui reprennent, par exemple, la liste des personnes de l'organisation et leurs droits d'accès à certaines informations. Pour accomplir cette fonction, un protocole issu du monde Internet convient parfaitement : LDAP (*Lightweight Directory Access Protocol*). C'est un protocole d'accès à une représentation arborescente d'un annuaire, dont les principales fonctionnalités sont la recherche d'une entrée, l'ajout ou la destruction d'une entrée, la modification d'une entrée et la comparaison entre deux entrées[37]. Nous

---

36 Tanenbaum cite également, au sein de la gestion de messagerie électronique, le protocole DMSP (*Distributed Mail System Protocol*) qui permet de gérer le courrier sur un serveur à partir d'un PC, même lorsqu'il est déconnecté du serveur.

37 Pour en savoir plus sur LDAP, cf. ALIN Fr., LAFONT D., MACARY J.-Fr., *op. cit.*, p. 159.

pouvons aussi citer, pour la gestion des annuaires, les solutions Whois++ (mécanisme issu de l'Internet qui ne fait pas autorité), X500 (norme ISO qui n'est pas respectée *de facto*) et NDS (protocole propriétaire de Novell).

I.3. Méthodes techniques de construction d'un intranet

Avec sa définition, l'infrastructure matérielle, logicielle et protocolaire qu'il requiert, nous connaissons les principes de fonctionnement d'un intranet. Mais il nous faut encore éclairer l'édification même du Web interne, c'est-à-dire aborder les méthodes de construction. Comment implémenter, concrètement, un intranet ?

Si un intranet se réduisait à de la publication et de la consultation d'informations, peu de techniques lui seraient nécessaires. Un réseau physique, un serveur Web, les protocoles de l'Internet et la présence d'un navigateur sur chacun des postes clients suffisent à la diffusion et à la réception d'information. Mais le concept d'intranet véhicule une idée de communication beaucoup plus puissante : il s'agit à la fois de distribuer et d'organiser, d'informer et de structurer. Autrement dit, l'information se doit d'être intelligemment fournie. Des applications d'information ou de gestion peuvent ainsi être développées au sein de l'intranet et être distribuées dans l'organisation, de manière à mieux gérer la communication. Nous allons voir dans cette section, comment la construction de ces nouveaux programmes est possible. De nouveau ici, nous devons distinguer le point de vue du poste client et celui du serveur.

I.3.1. Le poste client

Nous savons que les balises HTML permettent de mettre en forme les pages de texte à afficher au sein du navigateur. Mais les possibilités de HTML ne se réduisent pas à de la présentation d'informations sur un réseau. De véritables applications peuvent également y être déployées. Ainsi, au sein même des balises HTML, d'autres langages peuvent intervenir et permettre l'exécution de programmes, auxquels le navigateur vient simplement offrir son interface.

Du code « JavaScript » peut notamment être directement inséré au sein des balises HTML ; il sera alors interprété et exécuté par le navigateur. JavaScript[38] est un langage de script, c'est-à-dire un langage de programmation simple, développé par Netscape. Il permet d'introduire de nombreuses animations au sein des pages Web, de valider des informations envoyées vers un serveur ou de mettre en forme des résultats reçus de celui-ci. Par contre, JavaScript n'autorise pas une connexion avec un serveur distant. Une difficulté importante de ce langage réside dans les incompatibilités d'interprétation selon les navigateurs.

Outre JavaScript, on peut également utiliser le langage « Java ». Java n'est pas un langage de script, mais bien un langage orienté objet très évolué, très puissant et très complet, qui a été conçu par Sun. Des programmes Java compilés dans une forme intermédiaire, du « p-code » (*byte-code*) peuvent être directement appelés depuis du code HTML, chargés du

[38] La ressemblance de nom entre JavaScript et Java ne doit pas nous tromper. Il s'agit bien de langages totalement différents, conçus par des organisations hétérogènes. A en croire Frédéric Alin *et alii*, « le nom de JavaScript a été choisi uniquement pour des raisons marketing » (*op. cit.*, p. 176).

serveur vers le client et être ainsi exécutés à partir du navigateur, indépendamment de la plate-forme (ce sont des « applets » Java). Ces applets Java ont donc atteint un objectif presque mythique en informatique : la compatibilité entre les ordinateurs de types différents. Cependant, le caractère portable des applets doit être relativisé : le poste client doit disposer, d'une part de la machine virtuelle Java (*Java Virtual Machine*) chargée de lire le code intermédiaire et de le transformer en code exécutable, et d'autre part des interfaces de programmation standard en Java pour le système d'exploitation concerné (API – *Application Programming Interface*)[39]. De plus, si Java permet des développements très interactifs grâce à un environnement visuel orienté objet, le téléchargement de ces applets reste très coûteux en termes de temps d'accès et nécessite parfois des droits que la sécurité du système ne peut accorder (par exemple, l'écriture ou la lecture sur des disques durs locaux).

A l'instar d'une applet Java, un « contrôle ActiveX » peut également être exécuté à partir d'un document HTML. Un contrôle ActiveX est un programme compilé pour l'environnement Windows 32 bits, qui est préalablement enregistré sur le poste de travail de l'utilisateur. Dès lors, les contrôles ActiveX ont accès au disque dur du poste client ; ils ne doivent pas être téléchargés et les entrées-sorties sont plus rapides. La sécurité est assurée par des modules d'identification. Le point faible de ces contrôles ActiveX réside précisément dans l'atout de Java : ils sont limités à l'environnement Windows 32 bits... ce qui exclut à la fois les systèmes d'exploitation antérieurs à Windows 95 (c'est-à-dire la génération de Windows 3.11) et tous les systèmes étrangers à Microsoft. De plus, il faut être sûr que le poste client dispose des contrôles employés.

Grâce à JavaScript, à Java ou à ActiveX, les pages HTML d'un intranet peuvent donc rendre des services, offrir des applications à l'utilisateur[40]. Malgré sa simplicité et sa flexibilité, le protocole HTML souffre de défauts majeurs. HTML, en mêlant structure logique, balises de format et information à afficher, ne permet pas une maintenance aisée. Le langage XML (*eXtensible Markup Language*), qui utilise des feuilles de style, constitue sans doute une des solutions d'avenir à ce niveau.

I.3.2. Le serveur

Le protocole HTTP permet aux serveurs « passifs » de transférer des pages stockées en mémoire vers le poste client qui la demande. Mais, les serveurs « applicatifs », qui construisent une page toute spécifique suite à une requête du client, ne peuvent se contenter des fonctions inhérentes à ce protocole.

Ces serveurs requièrent un double mécanisme : d'une part il faut capter les paramètres de la requête client, d'autre part il s'agit d'exécuter la requête et de créer la page demandée.

[39] L'API (interface de programmation d'applications) consiste en près de 200 classes d'objets répartis en sept paquetages. Parmi ceux-ci, on trouvera les paquetages Java.awt, Java.awt.image, et Java.awt.peer, qui contiennent les méthodes de gestion des systèmes de fenêtrage de divers systèmes d'exploitation, et les événements correspondants. Pour que le p-code puisse tourner sur un système, il faut donc qu'il ait été prévu dans ces paquetages.

[40] La version 4 du langage HTML, intitulée « Dynamic HTML », propose des fonctions d'animation relativement élaborées. Nous ne l'avons pas traitée comme telle pour deux raisons. D'une part, ces animations ne peuvent être considérées comme des applications utiles à un intranet (elles rejoignent plutôt le rôle d'images animées, comme celles de format GIF) ; d'autre part, cette version du HTML n'est opérationnelle que sur les versions 4 des navigateurs Netscape et Internet Explorer, ce qui restreint considérablement sa portée.

Ce « dialogue » entre le serveur et le client peut être mis en œuvre de plusieurs manières. La solution la plus commune tient dans un CGI (*Commom Gateway Interface*) : il s'agit d'un ensemble de conventions qui permettent de traiter des données saisies dans un formulaire au sein d'une page HTML. Généralement, un CGI consiste dans des script déposés sur le serveur dans un répertoire spécifique (souvent appelé « cgi-bin ») et qui possèdent chacun une URL propre. Lorsqu'on envoie les données d'un formulaire HTML, le navigateur active le script correspondant à cette URL en lui passant les données du formulaire comme paramètres. Ce script peut effectuer de nombreux traitements (comme des interactions avec une base de données par exemple) et rendre au poste client des résultats, sous forme HTML. Ce transfert s'apparente alors à une simple méthode HTTP, comme dans le cas de serveurs statiques.

La solution CGI offre à la fois la simplicité et la souplesse : les scripts peuvent être écrits en n'importe quel langage, à condition toutefois que le serveur les supporte. Actuellement, les langages C et Perl sont les plus utilisés ; mais des scripts Shell sur une station Unix suffisent[41]. Le désavantage principal de cette solution consiste dans la consommation intense des ressources du serveur : pour chacune des requêtes du client, en effet, il faut de nouveau lancer un script du CGI.

Pour remédier à ce défaut du CGI, Microsoft et Netscape ont tous deux développé leur technique propriétaire. Sans les détailler, nous pouvons citer ISAPI – *Internet Server Application Programming Interface* – puis ASP (*Active Page Server*), pour Microsoft ; et, en ce qui concerne Netscape, NSAPI – *Netscape Server Application Programming Interface* puis LiveWire. Ces possibilités allient la puissance et la facilité d'emploi, mais elles demeurent la propriété exclusive de leur constructeur respectif : la maintenance et l'évolutivité sont restreintes.

A côté de ces solutions propriétaires de Microsoft et de Netscape, le langage Java, dont nous avons déjà souligné la puissance et surtout la portabilité sur tout type d'ordinateur, peut lui aussi gérer les serveurs. Les « servlets » sont des programmes Java fonctionnant en permanence sur le serveur et qui sont invoqués à la demande des clients. Dans la foulée de ces « servlets », on trouve également des *Java Server Pages*. Avec ces possibilités supplémentaires, le langage de Sun apparaît comme une voie d'avenir : « Les différents avantages de JSP (indépendance vis-à-vis du système d'exploitation et du serveur Web, facilité de mise en œuvre, réutilisabilité d'objets logiciels existants, qualité intrinsèque du langage Java) en font sans doute le meilleur candidat pour le développement des applications de taille moyenne ou importante »[42]. Vraisemblablement, le monde des intranets va bénéficier, avec l'association des applets et des servlets, d'outils de développement prometteurs.

Ainsi donc, les protocoles HTTP et HTML peuvent être complétés par des solutions très diverses. Nous avons très rapidement abordé les technologies CGI, ISAPI, NSAPI, ASP, LiveWire, applets ou servlets Java et JSP. Nous n'avons pas la prétention d'avoir cerné la question du dialogue entre un poste client et un serveur au sein de la voie intranet. Nous avons tout au plus posé les bases des techniques intranets, afin d'en éclairer les principes clés. Pour donner une vision complète de la construction d'un intranet actuellement, il nous faut encore évoquer la question de la sécurité.

---

[41] Tanenbaum n'aborde guère les scripts CGI. Pour en savoir plus sur ces derniers, cf. ALIN Fr., LAFONT D., MACARY J.-Fr., *op. cit.*, pp. 192-194.

[42] *Ibid.*, p. 200. Tanenbaum n'évoque même pas ces servlets. Pour en savoir plus, cf. *ibid.*, pp. 198-200.

I.3.3. La sécurité de l'intranet

Puisqu'un intranet concerne une communauté restreinte d'utilisateur, souvent ouverte sur l'Internet, le problème de la sécurité du Web interne se pose rapidement. Plus précisément, la sécurité sur un intranet est un double problème : à la fois externe et interne. D'une part, il s'agit d'empêcher des individus étrangers au système d'y entrer pour y consulter des informations, exécuter des applications ou causer des dégâts. D'autre part, il s'agit, au sein même de l'organisation, de veiller à ce que les droits et les pouvoirs de chacun soient respectés : il est souvent tentant de s'attribuer électroniquement des privilèges usurpés.

Pour régler le problème de la sécurité externe, la solution la plus opportune consiste sans nul doute dans un *firewall*. Il s'agit d'une passerelle d'accès (garde-barrière) entre l'Internet et l'intranet qui contrôle le trafic entre la Toile interne et le Web mondial, selon les règles qui ont été fixées a priori. Le *firewall* va instaurer un filtre au sein de l'accès à l'intranet : seules les personnes autorisées pourront y rentrer. La sécurité peut également limiter l'accès à certaines fonctions de l'intranet dans un sens : les employés peuvent par exemple surfer sur le Web, mais le serveur intranet ne peut être appelé de l'extérieur. Deux types de *firewall* existent : au niveau du réseau et au niveau de l'application. Le *firewall network level* fonctionne comme un routeur intelligent qui peut s'occuper de traduire les adresses internes du réseau en adresses IP publiques grâce au numéro du port client associé à chaque paquet de données. Le *firewall application level* est, quant à lui, un réel serveur proxy, qui peut analyser le trafic du réseau et exécuter des tâches complexes de filtrage et de sécurité[43].

Au niveau de la sécurité interne, plusieurs solutions sont envisageables. Par exemple, un système d'authentification peut être basé sur des mots de passe enregistrés avec les droits d'accès correspondants dans un annuaire. On peut aussi utiliser les *cookies*, qui sont des informations enregistrées sur le disque dur du poste client et qui sont automatiquement transmises par le navigateur. Dans le cas de configurations stables, on peut également aiguiller les utilisateurs dans l'intranet selon le numéro IP de la machine qu'ils emploient. Ces deux dernières solutions nécessitent toutefois que les mêmes ordinateurs soient accessibles aux personnes ayant les mêmes droits, ce qui signifie une certaine rigidité dans l'emploi du parc informatique. Il faudra donc adapter les possibilités de sécurité en fonction de la situation technique et organisationnelle.

Remarquons enfin, pour clore ce point sur la sécurité, qu'un chiffrement des données peut permettre une plus grande imperméabilité de l'intranet. Tous les algorithmes à clefs secrètes ou mieux, publiques, peuvent ici être employés. Comme nous l'avons déjà mentionné, un chiffrement de la couche transport est, quant à lui, prévu dans la nouvelle version d'IP, IPv6.

---

[43] Pour en savoir plus sur les principes des *firewalls*, cf. TANENBAUM Andrew, *op. cit.*, pp. 430-432, et cf. ESPLIN K., *art. cit.* Kathryn Esplin détaille six produits actuels de *firewall*, avec leur prix respectif (de 3000 à 25000 dollars US), la plate-forme sur laquelle ils peuvent être implantés et le niveau auquel ils s'appliquent.

I.4. L'architecture intranet

Si l'infrastructure matérielle, logicielle et protocolaire forme, en quelque sorte, la matière première d'un intranet, et si les méthodes de construction agencent cette matière première dans un produit semi-fini, nous devons désormais étudier comment le produit final peut être échafaudé. Autrement dit, c'est l'architecture concrète d'un intranet qui va nous occuper maintenant. Nous comprendrons ainsi comment la technologie et les principes peuvent prendre corps dans des solutions plus structurées.

I.4.1. L'architecture tripartite

De nombreux auteurs caractérisent une architecture intranet comme composée de trois niveaux : un client, un serveur applicatif et un serveur de données[44]. Grâce à cette architecture tripartite, le poste client ne doit pas interroger directement le serveur de données : entre les données et le client, le serveur applicatif vient jouer le rôle d'intermédiaire et ainsi les soulager du poids de l'infrastructure logicielle.

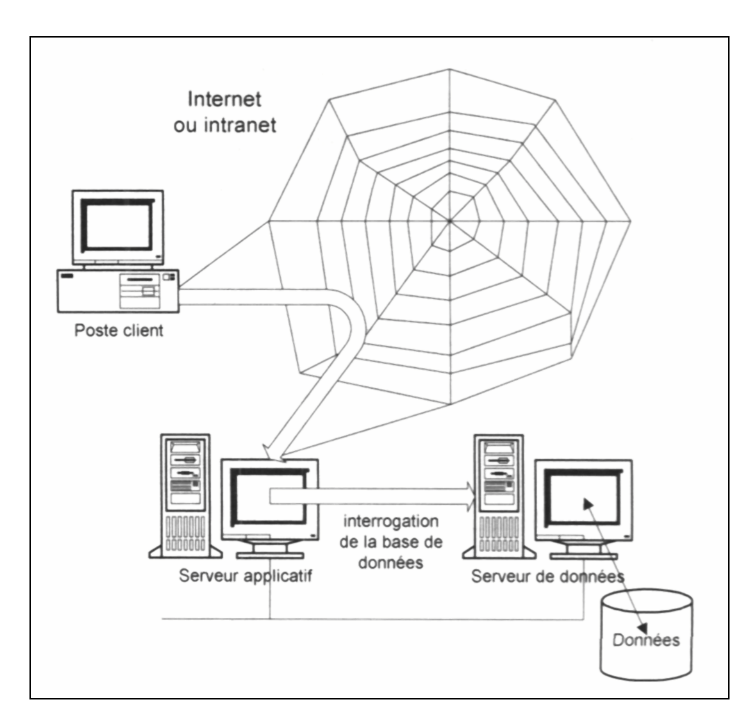


Figure 2 : Architecture tri-partite intranet
Issu de ALIN Fr., et al., *op. cit.*, p. 209.

Le client est un navigateur ou une machine virtuelle Java. Le serveur de données est un système de gestion de base de données relationnelles. Le serveur applicatif, quant à lui, met en œuvre la notion « d'objets métier », c'est-à-dire des objets qui peuvent relier plusieurs bases de données. Un serveur de ce type peut prendre essentiellement deux formes. Tout d'abord, le serveur applicatif peut se résumer à un serveur Web qui crée des pages HTML en réponse aux requêtes adressées à la base de données. Les outils employés par ce serveur ont déjà été présentés : CGI, ASP, LiveWire, les servlets… Dans ce cas, les protocoles HTTP et

[44] Cf. www.microsoft.com/france/intranet, et ALIN Fr., LAFONT D., MACARY J.-Fr., *op. cit.*, pp. 207-221.

HTML s'occupent de la transmission et de l'affichage des documents, issus des programmes concentrés sur le serveur : le navigateur sur le poste client joue simplement le rôle d'interface d'affichage. La deuxième forme possible d'un serveur applicatif consiste dans un serveur intranet client/serveur à proprement parler. Le serveur et le client possèdent alors leurs propres applications spécifiques. Sur le poste client, on trouvera surtout des contrôles ActiveX ou des applets Java. Les protocoles HTTP et HTML ne sont plus utilisés directement par ces applications : le navigateur appelle simplement l'application et lui fournit l'environnement de fonctionnement, mais l'affichage est pris en charge par les primitives de l'application et du navigateur.

I.4.2. Le middleware

Au sein de l'architecture intranet, un « *middleware* » permet la communication entre le client et le serveur : « Le middleware est donc la couche technique qui permet de transporter les données entre le client et le serveur [...] Plus précisément les couches middlewares assurent la conversion de données organisées (tables, structure, objets reliés entre eux), en données brutes, c'est-à-dire en caractères collés simplement les uns à la suite des autres »[45]. Le protocole HTTP constitue l'exemple le plus simple d'un *middleware* : pour un serveur Web, il est suffisant. Mais la simplicité du protocole HTTP va de pair avec ses inconvénients : HTTP est non structuré (ce sont uniquement des pages HTML qui sont transportées, et non des objets ou des messages), peu efficace (chaque document et chaque image à transmettre nécessite l'ouverture d'une connexion) et totalement incapable de gérer des sessions et des profils (pour conserver un environnement d'utilisation, il faut utiliser les *cookies* ou inclure des paramètres cachés dans les pages HTML ou garder des paramètres liés aux adresses TCP/IP sur le serveur). Pour des serveurs applicatifs, qui requièrent des *middlewares* plus complets, HTTP ne convient plus. D'autres solutions ont alors été élaborées.

Ainsi, les protagonistes des technologies orientées objet, rassemblés dans l'OMG (*Object Management Group*, une association d'éditeurs et de constructeurs), ont mis au point Corba (*Common Object Request Broker Architecture*). Corba est un ensemble de spécifications qui tentent de « définir comment un objet peut échanger des données avec un autre objet distant comme s'ils étaient sur la même machine »[46]. Les méthodes d'un objet situé sur le serveur sont accessibles par les objets du poste client, qui peuvent les appeler. Concrètement, Corba se compose de progiciels qui décodent et codent les demandes du client vers le serveur, et vice versa. Comme Corba a été développé en dehors de toute problématique liée à l'Internet ou aux intranets, il faut, pour l'utiliser dans ce cadre, le compléter par le protocole IIOP, qui définit comment deux progiciels Corba dialoguent en utilisant TCP/IP. La puissance de Corba provient surtout de son indépendance face à tout langage informatique et donc à tout environnement d'exploitation. Malheureusement, comme tous les outils universels et complets, Corba souffre d'une énorme complexité de mise en œuvre.

RMI (*Remote Method Invocation*) se propose alors comme un compromis entre l'universalité et la complexité. RMI a été construit par les concepteurs de Java : seuls des programmes Java peuvent donc communiquer avec RMI. Cette limite rend RMI beaucoup plus simple que Corba et même, à certains égards, plus puissant : en effet, là où Corba devait rester le plus général possible (à savoir dans le format des résultats ou des paramètres

[45] *Ibid.*, p. 211.
[46] *Ibid.*, p. 213.

transmis), RMI peut utiliser les normes Java et transmettre les résultats et les paramètres comme des objets eux-mêmes.

A côté du *middleware* de Sun, on trouve également DCOM (*Distributed Component Object Model*) de Microsoft. Il s'agit ici aussi d'une simplification de Corba, mais dédiée au monde Microsoft : plus précisément, ce ne sont plus des programmes Java qui peuvent communiquer avec DCOM, mais des contrôles ActiveX. Limité aux plates-formes Windows 32 bits, DCOM est moins puissant que Corba ou RMI, d'autant plus qu'il ne supporte pas l'héritage ; mais, au sein de ces plates-formes, DCOM peut permettre la communication entre des applications écrites avec des langages différents, ce que ne permet pas RMI.

Ces trois *middlewares* orientés objet constituent les solutions les plus répandues. Il existe aussi des *middlewares* orientés message : MQ Series d'IBM, Microsoft Message Queuing de Microsoft, Active Web d'Active Software… Comme ces outils ne sont pas encore normalisés, il nous est impossible d'exposer brièvement leurs principes de fonctionnement dans le cadre de cette section.

I.4.3. Le moniteur transactionnel

Pour épauler le *middleware*, les gros réseaux (pour plusieurs centaines ou plusieurs milliers d'utilisateurs en simultané) peuvent utiliser un « moniteur transactionnel », qui vient se poser comme un intermédiaire régulateur entre le serveur de données et le serveur applicatif. Le recours à cette solution n'est absolument pas obligatoire, mais il offre un confort important. Ce moniteur va, en quelque sorte, réguler le trafic des informations tout en assurant la cohérence de chacune des transactions entre les clients, le serveur applicatif et les bases de données. Lorsqu'ils sont parfaitement intégrés au *middelware*, ces outils sont rassemblés dans un OTM (*Object Transaction Monitor*). Vraisemblablement, « ces OTM constituent la clef de voûte des futures applications intranet de grande ampleur »[47].

Actuellement, ces OTM sont proposés par Microsoft, sous forme d'une architecture complète mais propriétaire, et par une association de nombreux industriels sous la forme d'une norme appelée « Enterprise Java Beans », indépendante de la plate-forme mais peu performante et très complexe.

Avec cette vision architecturale d'un intranet, construite à partir des fondements techniques que nous avons éclairés préalablement, nous disposons désormais des concepts clés du monde des intranets.

[47] *Ibid.*, p. 219.

I.5. Les services rendus par un intranet

De la notion d'intranet, nous connaissons désormais : sa définition, ses technologies, ses méthodes de mise en place concrète et son architecture. Mais nous ignorons encore ce qu'un intranet peut apporter. Certes, nous savons que c'est un outil de communication, qui permet de distribuer à la fois de l'information et des applications interactives, mais ces considérations restent trop générales. Pour aller plus loin, nous proposons de donner une vision plus précise des divers services qu'un intranet peut rendre à l'organisation dans lequel il est implémenté. Ces services peuvent être classés en deux catégories, conformément à la définition même d'un intranet : la distribution d'informations et la distribution d'applications.

I.5.1. La distribution d'informations

Au sein de l'organisation, l'intranet se veut tout d'abord le porte-parole de la voix officielle. On y trouvera l'information cautionnée par l'autorité de l'organisation, comme des documents importants, les communications officielles… Cela évite les frais d'édition de documents papiers (*reporting*) et la gestion lourde de panneaux d'affichage. Il faut particulièrement veiller à « définir et assembler intelligemment : du texte compréhensible par tous, de la secrétaire au président, des illustrations (images, schémas), qui agrémentent la lecture et accompagnent le discours, une navigation fluide et efficace entre les documents, pages et écrans »[48]. Par ailleurs, avec un intranet bien construit, l'information atteint les personnes concernées avec plus de certitude et, si la sécurité est bien assurée, seules ces personnes sont contactées.

L'information de l'intranet ne se réduit pas à l'officiel. Des espaces d'information libres peuvent aussi voir le jour. Des forums sur tel ou tel thème (*newsgroup*), des panneaux d'affichage électroniques pour les renseignements divers… peuvent être hébergés sur le Web interne. Au-delà de l'aspect socio-psychologique, la possibilité de publier de l'information officieuse permet de mieux diffuser des compétences et des renseignements que certaines personnes possèdent individuellement et qu'elles ne pourraient communiquer sinon : on imagine mal des valves consacrées à tel ou tel thème, que chacun pourrait consulter à souhait et où il pourrait déposer des questions ou des idées. En quelque sorte, ce sont les connaissances implicites de l'organisation qui peuvent ici trouver une mémoire et une publication.

Cette logique de rassemblement des connaissances implicites et explicites peut être prolongée. En effet, comme la section sur l'architecture nous l'a montré, un intranet est souvent relié à une ou plusieurs bases de données. Autrement dit, l'information qui passe sur l'intranet n'est pas nécessairement éphémère. Des mécanismes d'enregistrement et d'indexation peuvent permettre un archivage de ces renseignements électroniques. Ainsi, des serveurs de fichiers, de documents et de données peuvent assurer l'archivage et l'accès à l'information : des moteurs de recherche documentaire, des logiciels intermédiaires (*middelware*) assurant les requêtes SQL devront être associés au simple navigateur Web. Dès lors, il suffit d'un seul formulaire pour interroger l'ensemble des bases de données du système et tenter de trouver l'information que l'on souhaite. L'intranet peut ainsi devenir le canal d'entrée-sortie des bases de données réunies. Il est dès lors plus facile d'accéder à la mémoire de l'organisation, et aussi de contrôler sa sécurité et son intégrité.

---

48 *Ibid.*, p. 105.

Plus particulièrement, l'annuaire des membres de l'organisation, nécessaire pour la mise en place d'un intranet de grande envergure, peut se prêter à ce type de traitement. En effet, de nombreuses organisations publient régulièrement d'énormes catalogues avec leurs membres, leurs adresses, leurs rôles… On imagine sans peine la difficulté de mettre à jour ces données et les coûts d'impression et de reproduction que cela entraîne. La gestion électronique sur l'intranet de tels renseignements permet à chacun de corriger ses propres données personnelles et d'accéder, *on-line*, à celles d'autrui.

Enfin, l'intranet peut également permettre la récolte et la publication d'informations de contrôle et d'administration du réseau. Chaque connexion, puisqu'elle est reliée explicitement à un utilisateur, peut être suivie « à la trace ». Un espace spécifique sur l'intranet, uniquement accessible au *Webmaster* ou à son équipe, peut alors rassembler tous les renseignements sur les diverses connexions. En quelque sorte, l'intranet a cette particularité de pouvoir être contrôlé au moyen de ses propres outils.

I.5.2. La distribution d'applications

Comme nous l'avons vu, l'intranet peut aussi contenir des applications distribuées, grâce aux développements CGI, aux applets et servlets Java, aux contrôles ActiveX… Les services qu'il peut rendre ne se limitent plus, dès lors, à de la distribution d'information, même si le but poursuivi consiste toujours à « améliorer la communication et l'échange d'informations »[49]. Ces applications héritent ainsi des avantages du principe même de l'intranet : la distribution (telle une architecture client/serveur) et la standardisation (caractéristique de l'Internet).

Des applications de gestion peuvent ainsi être exécutées *on-line*, via le réseau interne. Si l'intranet est connecté à l'Internet, ces applications peuvent même être lancées de n'importe où dans le monde ! Pour des organisations dont les membres sont dispersés sur des zones géographiques éloignées, cette distribution des applications permet de centraliser toutes les opérations sur un même serveur, et les données dans des structures centralisées. Par ailleurs, une modification sur un programme ne nécessitera pas une mise à jour de chacun des postes clients, du moins s'il ne requiert pas un composant sur le disque dur de celui-ci (on pense aux contrôles ActiveX, par exemple). Le partage d'applications uniformisées permet dès lors de développer à l'infini les possibilités de l'intranet.

Des assistants de publication de l'information peuvent être intégrés dans l'intranet, comme des formulaires pour afficher des questions ou des idées sur les forums de discussion. Grâce à ces assistants de publication, les balises HTML ne doivent pas être connues des utilisateurs, du moins pour des services élémentaires. Quant à la production d'informations plus complexes, elle peut être effectuée grâce à des outils spécifiques (FrontPage de Microsoft, Netscape Composer de Netscape...) En intégrant tous les outils d'édition électronique, l'intranet devient véritablement « un système permettant d'organiser la production, la publication et l'administration de collections documentaires »[50].

Un véritable travail coopératif peut également être mis en place au moyen de cette infrastructure. Grâce au flux communicationel instauré par la distribution des applications et

[49] *Ibid.*, p. 121.
[50] *Ibid.*, p. 112.

des informations, les divers acteurs de l'organisation peuvent mieux coordonner leurs efforts dans la poursuite des buts communs. La messagerie, qui peut être complétée par un système de listes de diffusion, le partage d'un même document entre plusieurs personnes, des applications de rédaction simultanée, la gestion des agendas individuels et des ressources partageables, la visioconférence et l'audioconférence, les forums de discussion... constituent autant de fonctionnalités coopératives qu'un intranet peut offrir.

L'intranet peut même permettre la circulation de documents selon un schéma préétabli : c'est alors du *workflow*. Le *workflow* est un mode d'organisation du travail issu des systèmes d'information fondés sur les traitements. Ce type d'outils doit se fonder sur une formalisation extrême des procédures de l'organisation. Avec l'intégration du *workflow* sur un intranet, nous débouchons sur des systèmes d'information alliant les traitements et la communication c'est-à-dire sur des S.I. à la fois puissants et portables. Ce sont vraisemblablement les S.I. de demain, comme nous le supposions déjà dans notre première section en définissant le concept d'intranet. La distribution des procédures automatisées via le Web interne de l'organisation entraîne, en effet, à la fois une efficacité, une facilité et une homogénéité importantes dans la gestion courante des dossiers.

Bien sûr, en complément à tous ces programmes de gestion et de communication, des fonctions d'administration et de contrôle de l'intranet pourront, elles aussi, être implémentées.

I.6. Les impacts organisationnels potentiels

Les services que nous venons de distinguer constituent les buts recherchés explicitement par les acteurs d'un projet intranet. Cependant, introduire un intranet au sein d'une organisation ne se réduit pas à ajouter un outil informatique à la panoplie de ceux qui existent déjà. Nous voudrions montrer, dans cette section, qu'un intranet véhicule bien plus qu'un simple outil de communication électronique. En fait, c'est toute une perception de l'organisation qui est contenue dans la notion d'intranet : une perception novatrice… et prometteuse. Cette perception, parce qu'elle est implicite, ne s'impose pas : la mise en place d'un intranet ne signifiera pas nécessairement leur déploiement ; l'intranet vient comme proposer à l'organisation d'aller plus loin… libre à elle d'accepter ou de refuser.

Pour comprendre les conséquences organisationnelles qu'un intranet peut entraîner, nous avons découpé notre réflexion en trois niveaux . Cette distinction est toute personnelle : elle prête sans doute à la critique, mais elle nous semble éclairante. Un premier niveau serait celui de l'efficacité, c'est-à-dire de l'accroissement du rendement quantitatif des activités ; le deuxième, celui de la qualité, c'est-à-dire de l'amélioration qualitative des résultats du travail ; et enfin le niveau de la flexibilité, c'est-à-dire celui de l'augmentation de la capacité d'adaptation de l'organisation.

I.6.1. L'efficacité

Bien sûr, la première conséquence d'un intranet réside dans l'efficacité de la gestion de l'information. Sur le site de Radian Communication, on trouve le tableau suivant, qui reprend parfaitement les conséquences d'un intranet au premier niveau, celui du partage de l'information.

Sans Intranet	**Avec Intranet**
Production de documents coûteuse	Production de documents bon marché
Diffusion coûteuse (papier, photocopies, imprimerie, courrier, fax, ...)	Diffusion bon marché (réseau local, Internet)
Diffusion lente (courrier)	Diffusion rapide (téléchargement)
Obsolescence des documents	Mise à jour permanente des documents
Modèle basé sur la diffusion	**Modèle basé sur la demande**

Figure 3 : Tableau de comparaison de l'intranet et des systèmes traditionnels
Issu de www.radian.fr/intranet.html, Radian Communication 1999

I.6.2. La qualité

Au niveau de la qualité dans la gestion de l'information, les conséquences d'un intranet se situent davantage comme un approfondissement du mécanisme de communication et de coopération. Les organisations actuelles, en effet, sont souvent traversées par des flux d'informations non maîtrisés. Et les solutions électroniques, comme les messageries, sont souvent complices d'une telle surcharge informationnelle : « Information overload is a malady of our time »[51]. L'intranet, dans un flot toujours plus intense d'informations diverses, s'impose à la fois comme un moyen de faciliter la communication interne, comme une sécurisation des infrastructures et comme un lien vers les potentialités infinies de l'internet. Avec un intranet, une organisation gagne en rapidité de réaction, en dynamisme structural, en coordination de ses efforts. Un intranet contribue à passer de l'information « *just-in-case* » à l'information « *just-in-time* », c'est-à-dire à l'information « *on demand* »[52]. Par ailleurs, dans le chef d'une réflexion socio-organisationelle, un intranet peut permettre de trouver l'équilibre toujours recherché entre l'individu et l'organisation : en proposant à chacun une structure commune de communication et d'action, l'intranet offre à chaque individu la possibilité de recevoir, de découvrir l'identité collective, mais aussi le pouvoir de la construire, de s'y impliquer intensivement. L'intranet permet donc l'insertion passive (découverte de l'information) et active (publication de l'information) de chacun au sein du collectif : « L'égocentrisme dans l'entreprise laisse peu à peu la place au réseau-centrisme »[53].

I.6.3. La flexibilité

Cette structuration distribuée de l'information nous laisse présager des conséquences plus profondes encore. Certes, avec cette idée de « réseau-centrisme », nous constatons déjà que les intranets comportent une certaine mentalité organisationelle. Mais, plus abstraitement encore, c'est toute une théorie économico-politique qui est en jeu. Nous nous situons alors à un niveau plus symbolique.

---

51 TELLEEN St. L., *IntraNet Methodology. Concepts and Rationale*, *art. cit.*

52 Ces concepts proviennent de TELLEEN St. L., *IntraNet Methodology. Concepts and Rationale*, *art. cit.*

53 ALIN Fr., LAFONT D., MACARY J.-Fr., *op. cit.*, p. 34.

I.6.3.1. La situation organisationnelle

Ainsi, pour Steven Telleen, Directeur du Département Intranet chez Amdahl Corporation, un intranet peut amener des changements organisationnels profonds. Il contribue, en effet, à une meilleure « surface to volume ratio »[54]. L'analogie de la pomme de terre, qui cuit plus rapidement lorsqu'elle occupe moins de volume, illustre le propos de notre auteur. Selon lui, quand une organisation accroît son importance, sa complexité et son volume, elle devient plus lourde, plus inerte... Les décisions mettent beaucoup de temps pour passer du centre (le haut management) à la périphérie (les marchés, les clients, les départements). Lorsque l'inertie et le temps sont vaincus, l'information, enfin diffusée au sein de tous les membres de l'organisation, a souffert de nombreuses déformations et prête, dès lors, le flanc à des interprétations aussi personnelles que différentes.

La taille des organisations est donc, comme par nature, limitée : « This creates a natural limit to the size an organization can attain and still remain effective »[55]; « The overall result is a natural limit to the size organizations with a centralized decision-making structure can grow »[56]. Confrontées à leur propre développement, et donc à la croissance exponentielle de leur propre complexité, certaines organisations ont tenté de rationaliser leur structure, et donc d'éliminer une partie de leur personnel et de leurs services. Les cadres intermédiaires font souvent les frais de telles stratégies. Mais cette « rationale for downsizing » n'est pas réellement efficace. Les organisations peuvent aussi procéder à une restructuration complète de leur prise de décision et s'orienter vers une structure plus souple, plus « plate »... Mais, ces institutions sont suffisamment complexes pour ne pas être tranchées comme de simples pommes de terre : les prises de décision relèvent de processus éminemment diffus, qui ne peuvent être modifiés spontanément.

Pour surmonter ces obstacles, il nous faut tout d'abord éclairer les quelques principes qui guident désormais le développement et la gestion des organisations relativement importantes. Tout d'abord, les systèmes complexes sont plus transparents, plus souples et plus efficaces lorsqu'ils sont constitués de sous-systèmes autonomes, standardisés, auto-régulés. L'autorité a ainsi de plus en plus tendance à être distribuée: « The stressed organizations of today are already moving in this direction, reorganizing to move decision making from central, pyramidal, structures to distributed authority and decision making »[57]. Ce mouvement vers la décentralisation participe d'un « Business Process Reengineering »[58]. Dans une telle optique, la communication et la coordination s'imposent comme des éléments incontournables de la nouvelle structure de l'organisation. Mais, nous « have been missing was an appropriate infrastructure to support the communication and coordination requirements of these new decentralized organizations »[59]. Cette nouvelle infrastructure de communication et de coordination, adaptée aux formes actuelles organisationnelles, doit être inventée par la technologie : ce sera, on s'en doute, l'intranet.

---

54 TELLEEN St. L., *Intranets and Adaptive Innovation: The move from control to coordination in today's organizations, art. cit.*

55 *Ibid.*

56 *Ibid.*

57 *Ibid.*

58 *Ibid.*

59 *Ibid.*

I.6.3.2. La solution technologique

Aujourd'hui, l'outil de communication le plus commun et le plus puissant est sans conteste le Web. C'est en s'inspirant des potentialités et des facilités de l'Internet que la technologie intranet a pu voir le jour et répondre aux besoins actuels des entreprises et des services publics. L'intranet s'impose désormais comme la possibilité facile, pour les organisations, de supporter à la fois les prises de décision décentralisées et la coordination des diverses activités. Un intranet répond aux trois critères des organisations actuelles, comme mentionné ci-dessus : la limite naturelle due à la « surface to volume ratios »[60], l'auto-régulation et l'autonomie des sous-systèmes du système, la communication et la coordination au sein de l'institution.

Le rôle de l'intranet peut être expliqué à la lumière de théories économiques contemporaines, comme par exemple, celle de Michael Rothschild (auteur notamment du livre *Bionomics* paru en 1990) ou celle de James Moore (qui a écrit, par exemple, *The Death of Competition* en 1996). Ces auteurs utilisent tous deux l'analogie biologique de l'évolution pour expliquer le fonctionnement socio-économique. Pour Rothschild, la compétition constitue la dynamique fondamentale de l'évolution ; alors que Moore, pour sa part, élabore plutôt le concept de « coévolution » pour rendre compte de la complexité des écosystèmes et de leur développement. Dès lors, l'économie, pour chacun de ces théoriciens économistes, se comprend différemment : pour Rothschild, le système économique est essentiellement chaotique et, pour Moore, il est dirigé par des principes. Cette distinction entre « chaotic system » et « purpose-driven system »[61] est fondamentale pour comprendre les modèles de communication : les systèmes chaotiques correspondent aux systèmes distribués de communication et les systèmes dirigés aux systèmes distribués de prise de décision.

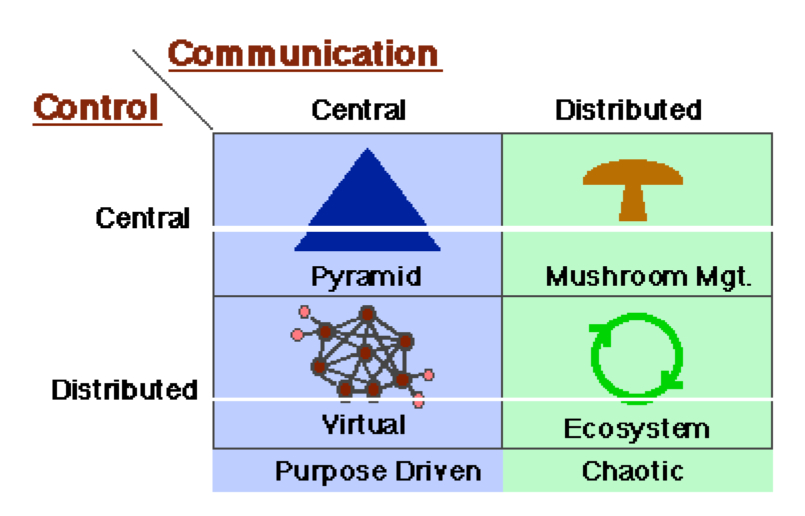


Figure 4 : Les modèles organisationnels
Issu de TELLEEN St. L., *Intranets and Adaptive Innovation…*, *art. cit.*

60 *Ibid.*

61 *Ibid.*

Lorsque les organisations deviennent trop importantes, le modèle pyramidal ne possède plus la même force opérationnelle. Le système devient chaotique, car les décisions subissent à la fois l'assaut du temps et des interprétations personnelles dans leur parcours du haut de la pyramide vers le bas. Le système distribué du contrôle et de la prise de décision, quant à lui, semble plus adapté pour des organisations de grande taille. Ce modèle peut prendre deux visages. D'une part, il peut garder une structure centralisée de communication et de coordination : c'est l'organisation des systèmes complexes, sur base de laquelle les technologies internet se sont développées. D'autre part, un système décentralisé peut s'appuyer sur une division en sous-systèmes autonomes et auto-régulés. Le développement organisationnel est ici plus chaotique[62], et se rapproche du fonctionnement naturel d'un écosystème. Cette division du système permet une ouverture constante à l'innovation et à l'adaptation: « I call the process that drives the distributed, decision-making and control systems adaptative innovation »[63].

Entre le modèle distribué (« the completely distributed (chaos) approach ») et le modèle centralisé (« the completely centralized (pyramid) approach »), le choix s'avère impossible. Ni le chaos ni des principes rationnels ne peuvent totalement diriger une organisation. Aujourd'hui, il faut trouver un compromis entre ces modèles: « a distributed management approach can be reconciled with purpose-driven results »[64].

L'intranet permet de trouver ce consensus. L'outil de communication devient ainsi un moyen pour obtenir une organisation plus adaptée aux temps contemporains. C'est le pari qu'a osé relever la technologie intranet : « The challenge is meating the needs for cordination and efficiency without destroying the independence of decision making and action that make entreprise strong and flexible. »[65]. Bien sûr, comme tout outil, les intranets peuvent être détournés de ce but qui, en quelque sorte, leur « colle à la peau ». Mais, si on laisse la dynamique qui les habite se développer, on devrait déboucher sur une organisation plus fluide et plus rationnelle, sur un mélange entre souplesse et rationalité, sur une organisation cohérente mais ouverte.

---

62 Ce mode de fonctionnement plus chaotique n'est pas sans rappeler ce que des sociologues appellent « le *garbage can model* », cf. COHEN M. D., MARCH J. G., OLSEN J. P., *A Garbage Can Model of Organizational Choice*, in *Administrative Science Quarterly*, vol. 17, 1972, pp. 1-25.

63 TELLEEN St. L., *Intranets and Adaptive Innovation: The move from control to coordination in today's organizations*, *art. cit.*

64 *Ibid.*

65 TELLEEN St. L., *IntraNet Methodology. Concepts and Rationale*, *art. cit.*

Conclusion du chapitre I

Dans ce premier chapitre, nous avons éclairé les principes de base des intranets. Rappelons les acquis de notre parcours avant d'aller plus loin.

Pour tenter de définir un intranet, nous avons tiré les quelques lignes de force qui se dégagent, actuellement, de ce concept : l'intranet est une utilisation des technologies et des moyens de l'Internet, dans le cadre d'une organisation particulière, pour faciliter la communication et même l'installer au centre du système d'information. Afin d'aller plus loin, nous avons alors examiné les composants essentiels de tout intranet pour les assembler petit à petit dans des solutions organisationnelles de haut niveau.

L'infrastructure nécessaire à un intranet est directement héritée de celle de l'Internet. Le matériel tout d'abord se réduit à un réseau local ou distant et à un serveur. Au niveau logiciel, le poste client ne requiert qu'un navigateur (*browser*), même si d'autres programmes peuvent rendre des services appréciables (comme des gestionnaires de messagerie électronique, par exemple). Sur le serveur, un logiciel doit traiter les requêtes du poste client : pour des requêtes complexes, des logiciels sur mesure seront nécessaires ; pour des cas simples, un programme d'envoi de pages Web suffira. Plus profondément, le matériel et le logiciel fonctionnent grâce à toute une série d'outils fondamentaux : les protocoles. Tous les protocoles des intranets proviennent de l'Internet : IP s'occupe d'orienter (routage) les segments de données dans les nœuds du réseau (couche réseau), TCP ou UDP se charge de la transmission correcte des données (couche transport), HTTP veille aux transferts des pages du serveur vers le client (couche application), HTML se préoccupe de la mise en page des informations sur le poste client (couche application), SMTP et POP3 ou IMAP4 gèrent les échanges de courrier électronique (couche application), LDAP se consacre au fonctionnement des annuaires (couche application)…

Tous ces éléments techniques constituent, en quelque sorte, la mécanique de base d'un intranet. Mais il faut encore agencer ces éléments dans des entités plus avancées : il s'agit de construire l'intranet afin de permettre à la fois la publication électronique d'informations et la distribution d'applications interactives. La construction de l'intranet peut être réalisée sur deux niveaux : le client et le serveur. Le poste client peut recevoir du serveur des pages HTML avec éventuellement du code JavaScript, charger des applets Java, exécuter des contrôles ActiveX ; ce sont alors les ressources du client qui sont utilisées. Le serveur, pour sa part, peut utiliser un CGI, employer des technologies propriétaires, lancer des servlets Java ou JSP, pour ensuite envoyer au client le résultat de l'exécution. Quant à la sécurité de l'intranet, elle pourra être assurée au niveau externe par un garde-barrière (*firewall*) et au niveau interne par l'emploi d'un annuaire et de mots de passe, par l'appel de *cookies* ou par un tri sur le numéro IP de la machine.

Avec ces éléments de construction, l'intranet ne forme encore qu'une boîte à outils. Ces derniers ne sont pas encore organisés dans une architecture spécifique, qui leur conférera le statut d'outil de communication organisationnel. L'architecture type d'un intranet se caractérise par trois composants : le poste client, le serveur de données, qui emploie souvent un gestionnaire de base de données, et le serveur applicatif, qui joue le rôle d'intermédiaire entre le client et les bases de données. La couche technique qui permet le transfert des données s'appelle un *middelware* : HTTP suffit pour les cas simples, Corba, RMI ou DCOM s'imposent si les applications présentent une complexité trop importante. Les gros réseaux

emploient aussi un moniteur transactionnel pour gérer le trafic des informations et contrôler la cohérence des transactions.

Ainsi mis en place, l'intranet peut jouer son rôle. Les services qu'il rend tournent toujours autour de la publication d'informations et du partage d'applications interactives. Ainsi, l'intranet peut permettre la publication de l'information officielle et de l'information officieuse et constituer le canal de transmission des connaissances et des compétences personnelles des membres de l'organisation. L'intranet peut aussi constituer le goulot d'accès aux bases de données centralisées. Lorsqu'il distribue des applications, l'intranet peut déployer un système complet de documentation, mettre en œuvre des structures de travail coopératif et même agencer du *workflow*.

Ces services sont directement impliqués par l'intranet, mais ce dernier peut également amener des conséquences indirectes dans l'organisation. La distribution de l'information au sein du Web interne peut, en effet, augmenter l'efficacité et la qualité des activités. Mais, plus indirectement encore, c'est la structure organisationnelle elle-même qui peut subir des modifications. Alors que la taille croissante des institutions actuelle apparaît de plus en plus comme un obstacle à la circulation de l'information, un intranet permet d'abandonner un modèle pyramidal de répartition du pouvoir pour une structure plus flexible, le modèle de la prise de décision distribuée.

Derrière le mot intranet, c'est donc tout un monde qui semble se cacher. Nous avons pu, dans ce chapitre, en percevoir les composants techniques, les méthodes de construction, l'architecture type, les services offerts et les conséquences organisationnelles potentielles. Bien sûr cette vision n'est complète. Néanmoins, elle donne le cadre théorique suffisant pour comprendre les choix et enjeux qui sous-tendent l'implantation d'un intranet dans une organisation, en l'occurrence l'Institut d'Informatique.

Page blanche

Chapitre II. Un intranet à l'Institut

Le premier chapitre nous a montré en quoi consiste un intranet : sa définition, l'infrastructure qu'il requiert, les méthodes de construction qu'il appelle, l'architecture dans lequel il s'incarne, les services qu'il rend et les conséquences organisationnelles qu'il peut impliquer.

Désormais, nous allons nous efforcer de poser les bases d'un intranet concret : celui de l'Institut d'Informatique des Facultés Universitaires Notre-Dame de la Paix. Pour atteindre cet objectif, nous exposerons tout d'abord quelques éléments théoriques issus de l'expérience de consultants dans le domaine d'implantation d'intranets. Ensuite, nous examinerons ce qui existe déjà à l'Institut en termes d'intranet. A ce stade, nous pourrons nous pencher sur les besoins non encore satisfaits à l'Institut, pour ensuite les évaluer et les classer selon leur priorité.

II.1. La mise en place d'un intranet : éléments de théorie

II.1.1. Quelques principes généraux

Les intranets, en s'attaquant à la distribution de l'information et des applications, se situent au cœur de la structure fonctionnelle de l'organisation. La concrétisation d'un intranet relève, dès lors, d'un processus risqué. Le déploiement d'un tel outil ne peut être laissé au hasard : les quelques expériences déjà vécues en la matière ont donné à des consultants certains jalons théoriques.

Ainsi, les auteurs du livre *Le projet intranet*, qui sont consultants chez Fi System en France, distinguent quatre principes généraux, qui doivent absolument sous-tendre la mise en œuvre de tout intranet.

Tout d'abord, les composants du système d'information doivent être autant que possible standardisés. Il faut un système d'information modulable, qui ne s'enferme pas dans des technologies trop spécifiques. Mieux vaut travailler avec les standards du marché, qu'on organisera en briques interchangeables, plutôt qu'avec des solutions trop pointues et trop particulières dont l'évolutivité et la portabilité seraient liées au (seul) constructeur. Puisque l'intranet se situe dans le monde mouvant de l'Internet, il faut impérativement garder la possibilité d'une « évolution progressive d'un système d'information à géométrie variable »[66].

Cette souplesse du système d'information doit aller de pair avec un recentrement de ce dernier vers les serveurs : l'administration du système d'information doit être située sur le serveur, et non déléguée sur les postes clients. Ici, il faut distinguer « distribuer » de « déléguer ». Un intranet permet certes de « distribuer » les informations et les applications, mais à partir d'un poste centralisé : le serveur. Les postes clients accèdent à celui-ci pour charger les renseignements et les applications, et même si certaines applications sont situées sur le poste client, la manière dont elles sont employées sera dictée par le serveur. Cette centralisation des informations et des applications sur le serveur est la condition nécessaire

[66] ALIN Fr., LAFONT D., MACARY J.-Fr., *Le projet intranet. De l'analyse des besoins de l'entreprise à la mise en œuvre de solutions*, Paris, Eyrolles, Coll. Fi System, 1998, p. 51.

d'une distribution cohérente et uniforme parmi tous les hôtes du réseau, en même temps que d'un contrôle et d'une gestion plus efficaces de l'information.

Cette centralisation de l'administration de l'information ne doit pas nous détourner du but final de l'intranet : un service de communication destiné aux membres d'une organisation. Dès lors, tous les développements de l'intranet doivent être dirigés en fonction de ces membres, les utilisateurs. Nous avons déjà vu combien l'interface du client universel, le navigateur, permettait un apprentissage rapide et un usage aisé de toutes les fonctions offertes dans l'intranet. Les responsables de l'intranet veilleront, de plus, à donner beaucoup de satisfaction aux utilisateurs, de manière à ce qu'il soit alimenté par les informations qu'ils possèdent personnellement, et qu'ils ont plaisir à diffuser et à recevoir.

Le lien entre la communication et l'organisation nous amène au dernier principe de développement d'un intranet : le système d'information doit être orienté vers la communication. En effet, les systèmes d'informations de ces vingt dernières années ont surtout été construits autour des « traitements », dans le but de favoriser la production de masse. Ces systèmes corroboraient souvent une organisation très pyramidale, où la communication ne représentait pas une donnée critique. Un intranet risque de bouleverser cette vision, comme nous l'avons vu au terme du chapitre précédent : le système d'information, lorsqu'il est complété par un intranet, n'est plus seulement orienté vers les traitements, mais aussi et surtout vers la communication. Mettre en place un intranet, c'est donc accepter l'avènement de changements structurels, c'est-à-dire la révision d'habitudes et de procédures au sein de l'organisation : messageries électroniques, forums, et autres moyens de communication ne peuvent être omis par peur du changement.

Ces quatre principes de base constituent en quelque sorte les pré-requis indispensables à l'avènement d'un Web interne : sans une standardisation, une centralisation, une priorité donnée à l'utilisateur et une orientation vers la communication, un intranet n'a pas de sens… Mais, bien sûr, ces principes généraux ne constituent pas une condition suffisante à la mise en place de l'intranet. Il faut encore prolonger ces principes par une conduite de projet très rigoureuse.

II.1.2. Le projet intranet

Comme dans tout projet aux conséquences organisationnelles potentiellement lourdes, la préparation semble bien plus importante que le déploiement concret. Distribuer à la fois les informations et les applications au sein d'un outil de communication n'exige pas seulement une maîtrise de diverses techniques de pointe et de recettes concrètes de mise en œuvre. Plus profondément, les responsables d'un projet intranet devront s'imprégner de la structure de l'organisation afin de discerner les besoins pour les traduire de manière adéquate.

II.1.2.1. La préparation

Selon Steven Telleen, la préparation d'un projet intranet devra s'effectuer en trois analyses : l'analyse politique, l'analyse technique et l'analyse du contenu (analyse fonctionnelle) ; « Constructing an effective Intranet infrastructure requieres attention to three distinct areas : management, technical, and content. Management consists of the roles, policies, processes and organization needed to manage the life cycle of formal Intranet

content. The technical infrastructure consists of the networks, hardware and software required to support content development, publishing and access. And, content requires processes to be developed to support special needs such as initial conversions, creation of database or application interfaces and development of 'glossy' pages for high impact »[67]. Parcourons chacun de ces trois pôles d'analyse de la phase de préparation de tout intranet.

L'aspect politique est bien sûr un élément difficile à maîtriser, car il implique souvent des aspects implicites et diffus. Il importe non seulement de bien comprendre l'organigramme hiérarchique de l'organisation, avec la distribution de pouvoir qu'il représente, mais aussi de tenir compte des jeux d'influence, des pôles de compétence et des forces de pouvoir qui existent *de facto* sur le terrain. Pour atteindre cette prise de conscience des « coulisses » organisationnelles, seul un contact approfondi (en qualité et en temps) avec les uns et les autres permettra de mettre à jour les caractéristiques officieuses de l'organisation. Dès lors, le management d'un intranet ne peut se contenter d'un « *Webmaster* » technique, comme dans le monde Internet habituel. Le concept d'intranet ouvre la voie à une autre compréhension du management : il requiert d'avantage un « IT administrator »[68], c'est-à-dire un gestionnaire de l'information. Ce dernier doit veiller à la publication, à la distribution, à la réception, à l'indexation et à la structuration de l'ensemble du flux communicationnel de l'organisation, en tenant compte de ses structures politiques. Bien souvent, dès lors, on met sur pied une équipe « *Webmaster* », un « *IT staff* », qui gère l'ensemble de l'intranet.

L'analyse des aspects techniques, deuxième phase d'analyse de la préparation d'un intranet, consiste à examiner successivement les besoins d'infrastructure, de serveur Web, de navigateur (*browser*) et de garde-barrière (*firewall*). Pour chacun de ces points, il importe de mesurer le plus précisément possible les capacités nécessaires, sans exiger de ressources inutiles. Au départ, on veillera à tenir compte des évolutions possibles des besoins de l'intranet : il est évident que l'infrastructure matérielle ne peut changer constamment. L'analyse technique s'efforcera donc de distinguer les éléments dont la puissance peut facilement être accrue, de ceux qui sont peu évolutifs. Pour les premiers, on pourra se satisfaire des besoins énoncés au départ, mais pour les autres, il faudra entreprendre une analyse plus approfondie : le réseau, par exemple, risque-t-il de devoir supporter de la visioconférence à un moyen terme ?

Pour le troisième pôle d'analyse de la préparation du projet intranet, à savoir l'analyse fonctionnelle (ou analyse du contenu), il faut s'interroger sur les éléments à implémenter sur l'intranet, c'est-à-dire les fonctions, les compétences et le pouvoir distribuables. Pour mieux cerner les applications à développer au sein de l'intranet, une étude des besoins devra être opérée, avec un examen minutieux des moyens possibles de réalisation (CGI, applets Java…). Comme l'intranet est destiné à des modifications continuelles, son contenu devra être très structuré, de manière à faciliter toute mise à jour.

---

[67] TELLEEN St. L., *The IntraNet Architecture: Managing information in the new paradigm*, Copyright © 1996 Amdahl Corporation, Sunnyvale, California, USA, cf. http://www.amdahl.com/doc/products/bsg/intra/infra.htm.

[68] L'expression « *IT administrator* » vient de Kathryn Esplyn, cf. ESPLIN K., *8 important issues to consider before building an intranet, What must you know about your intranet's infrastructure and staffing to be successful? These guidelines will help you through the morass of intranet planning. Plus Sun and Lockheed Martin tell how they constructed their massive intranets*, Mars 1997, cf. www.sunworld.com/swol-03-1997/swol-03-intranet.html.

Ces trois pôles d'analyse, qui structurent la phase de préparation du déploiement d'un intranet, peuvent s'avérer essentiels à la réussite du projet intranet. Dans la foulée de la préparation du projet, la réalisation des premières briques de la Toile interne sera elle aussi cruciale : à ce stade, il importera de respecter scrupuleusement les résultats de l'analyse tripartite et de traduire les besoins des utilisateurs dans des outils efficaces et faciles. Ces premières étapes peuvent alors s'inscrire dans un développement plus large de l'intranet, qui avancera par incréments successifs.

II.1.2.2. Le développement incrémental

Dans un projet intranet, les utilisateurs occupent une place prédominante : ce sont surtout eux qui fournissent l'information à diffuser et qui la reçoivent. La réussite du projet intranet dépend donc de l'implication des utilisateurs, de la prise en compte de leur besoin et d'une utilisation aisée de l'outil. Pour « coller » aux utilisateurs et pour rassurer les divers acteurs du projet, une démarche par prototypage, avec des fonctions réduites et une envergure amoindrie, permettra à chacun de diminuer les risques d'échec : « The basic managment framework can be implemented quickly using our supplied templates, prototypes, and design models »[69]. Mais placer les utilisateurs au centre du projet ne suffit pas ; les techniciens et les informaticiens devront veiller à l'efficacité de l'intranet constitué. Il faut que l'outil soit effectivement opérationnel, tout en répondant aux besoins de chacun : « La clé de la réussite du projet consistera donc à rechercher un partenariat dynamique entre maîtrise d'ouvrage et maîtrise d'œuvre »[70].

Selon Microsoft, trois grandes étapes peuvent être distinguées au sein de ce développement incrémental. La première et la plus simple est celle qui concerne la création et le partage d'informations. Des outils de création de pages HTML, de mises à jour automatiques de pages de liens, des moteurs de recherche d'information… pourront ici être implémentés. La deuxième étape met en œuvre tout ce qui doit faciliter le travail en collaboration : une messagerie, des forums de discussion (*newsgroup*), des annuaires… Enfin, les applications de gestion seront mises en œuvre dans la dernière étape ; des programmes plus complexes, avec des gestionnaires de base de données spécifiques, pourront alors être créés via un CGI, les langage Java ou JavaScript, les contrôles ActiveX…

II.1.2.3. L'évaluation continue

Pour atteindre « l'objectif intranet », l'organisation doit avoir une conscience suffisamment éclairée d'elle-même. Autrement dit, les champions du projet intranet doivent toujours garder à l'esprit la stratégie de l'organisation, ses structures, sa hiérarchie, ses modes de fonctionnement, ses principes généraux, sa philosophie, sa culture. L'outil de communication qui est prévu doit être en accord avec l'organisation qui le porte. Dès lors, une évaluation continue du développement de l'intranet doit être effectuée.

---

69 TELLEEN St. L., *IntraNet Methodology. Concepts and Rationale*, Copyright © 1996 Amdahl Corporation, Sunnyvale, California, USA, cf. http://www.amdahl.com/doc/products/bsg/intra/concept.htm.

70 ALIN Fr., LAFONT D., MACARY J.-Fr., *op. cit.*, p. 249.

Sept questions peuvent accompagner ce développement incrémental et aider à l'évaluation continue du projet. Elles « devront trouver une réponse appropriée au bon moment :

- *De quoi* ont besoin les utilisateurs, quelles sont les cibles à atteindre ?
- *Pour quoi* faire, pour répondre à quels objectifs et à quelles attentes ?
- *Comment* satisfaire ces objectifs, avec quels moyens, quelles fonctionnalités sont nécessaires, quelle architecture peut répondre au besoin ?
- *Qui* participe au projet, qui conçoit, qui réalise, qui exploite, qui actualise?
- *Où* sont installées les ressources nécessaires ?
- *Quand* doit-on mettre en œuvre, quel est le planning, quelle est la stratégie de déploiement ?
- *Combien* cela coûte-t-il, quels retours sur investissement peut-on espérer ? »[71].

Il n'est pas exclu que la rationalisation de l'information, via le projet intranet, amène à interroger l'organisation quant à son infrastructure, ses modalités structurelles, les rôles fonctionnels et les degrés de pouvoir. A ce stade, les risques politiques sont bien sûr prépondérants. Puisqu'il s'agit de technologies relativement récentes et d'implications organisationnelles importantes, le projet intranet doit, en principe, être toujours considéré comme « risqué ». Dès lors, tous les moyens de diminution du risque habituels doivent être mis en œuvre.

II.2. L'existant à l'Institut

Ces quelques jalons théoriques sur la mise en œuvre d'un intranet attirent notre attention sur l'analyse préalable de la situation à effectuer : les pré-requis d'un intranet ont-ils déjà été établis, comment le contexte peut-il être décrit politiquement, techniquement et fonctionnellement, quels sont les utilisateurs et les besoins qui entourent le projet intranet… Ces questions doivent, à l'aube de notre projet d'intranet à l'Institut, trouver une réponse.

Or, la construction d'un intranet à l'Institut n'est pas une *creatio ex nihilo*. Quelques éléments d'intranet existent d'ores et déjà. Nous nous proposons, dans cette deuxième section du deuxième chapitre, d'analyser cet existant au moyen du cadre théorique que nous avons posé[72].

II.2.1. Description

L'Institut d'Informatique possède déjà un réseau interne, avec sa bande passante, ses serveurs, ses postes clients, ses protocoles (dont TCIP/IP). Deux *browsers* sont disponibles sur les machines clientes de ce réseau : celui de Netscape (« Navigator 4.0 ») et celui de Microsoft (« Internet Explorer 4.0 »). Un logiciel de messagerie électronique existe déjà, lui aussi : il s'agit d'« Eudora » (versions Light et Pro).

---

71 *Ibid.*, p. 251.

72 Pour obtenir une première vision du but poursuivi par l'installation d'un réseau à l'Institut, cf. GOOSSENS P., *Nouvelles Utilisations Réseaux,* Namur, F.U.N.D.P. - Institut d'Informatique, document interne.

Un site Web est déjà opérationnel tant pour les Facultés que pour l'Institut. Sur le site de l'Institut, on trouve même une ébauche d'intranet. Ce dernier est réservé aux membres du staff. Quelques informations et quelques services y sont disponibles : la réservation de locaux et de matériel. Ces services sont implémentés grâce à un CGI : des formulaires spécifiques permettent de capturer les requêtes des utilisateurs, qui sont envoyées par HTTP à des scripts Unix sur le serveur, qui effectuent le traitement *ad hoc* et renvoient une confirmation du résultat (ou un message d'erreur) au poste client.

Pour les étudiants, il existe également un espace Web qui leur est consacré, où ils peuvent trouver des informations les concernant : cet espace informatif a autrefois porté le nom « d'intranet des étudiants », pour aujourd'hui s'appeler « Informations pour les étudiants ». On y trouve des informations sur la bibliothèque des Facultés, des directives pour la rédaction d'un mémoire…

Par ailleurs, le site Web de l'Institut propose également la liste de ses membres : une liste reprend les membres du staff, et une autre les étudiants en les classant selon leur année d'étude. Chaque liste indique, pour chacun, l'adresse électronique qui lui correspond. En quelque sorte, il s'agit ici d'un annuaire élémentaire, non géré par LDAP, Whois++ ou toute autre technologie de pointe, car sa taille est relativement réduite.

La sécurité de ces quelques fonctions est d'ores et déjà assurée par un système simple, mais efficace. D'une part, un *firewall* (situé au Centre de Calcul) empêche certaines manipulations (par exemple, les étudiants ne peuvent envoyer du courrier électronique via le serveur SMTP de l'Institut qu'à partir d'une machine reprise dans le parc informatique des Facultés). D'autre part, un filtre est effectué sur les adresses IP des machines qui tentent d'accéder à l'espace « intranet » consacré aux membres du personnel : tous les ordinateurs réservés aux étudiants sont ainsi empêchés d'accéder aux services du staff. Une vigilance est donc exercée tant au niveau extérieur que du point de vue interne.

II.2.2. Analyse

Cette brève description de l'infrastructure téléinformatique de l'Institut nous montre que certains éléments d'un intranet sont déjà présents. Mais il faut désormais nous interroger sur la valeur de cet acquis : s'agit-il d'un fondement intéressant pour la mise en œuvre d'un intranet comme tel, ou d'un point de départ bancal ? Utilisons les éléments théoriques présentés au départ de ce chapitre pour répondre à cette question essentielle.

II.2.2.1. Les pré-requis de l'intranet

Tout d'abord, nous pouvons remarquer que les principes généraux nécessaires au déploiement d'un Web interne ont d'ores et déjà été appliqués. En effet, en regroupant les informations et les services utiles au staff et aux étudiants sur des espaces virtuels, accessibles depuis son site Web, l'Institut a entrepris un double effort de standardisation et de recentrement de ses ressources informationnelles. D'une part, les informations et les services sont présentés de manière homogène (sur des pages Web de format semblable), et sont implémentés de façon similaire (via un CGI écrit en scripts Unix). D'autre part, toutes les données sont rassemblées sur le même serveur. Bien sûr, d'autres données et d'autres

applications existent en dehors de cette structure Internet/intranet, mais l'effort de centralisation homogène doit être souligné.

Dans la foulée de cet effort, nous pouvons affirmer qu'une priorité a été donnée à l'utilisateur : toutes les fonctionnalités sont simples d'emploi et convainquent par leur ergonomie conviviale. De plus, la structure des pages Web reflète celle des utilisateurs, puisque les étudiants sont bien séparés des membres du staff.

Pour en terminer avec cet examen des principes généraux, nous pouvons encore remarquer que les outils déposés sur l'ébauche d'intranet contribuent à une communication toujours plus améliorée : tant l'espace destiné aux étudiants que celui réservé aux membres du staff n'ont d'autre but que de communiquer des informations. Et, en ce sens, les applications déjà opérationnelles actuellement, à savoir la réservation de matériel et celle de locaux, permettent en fait d'établir une communication efficace sur des aspects de gestion spécifique de l'Institut.

Puisque ces principes généraux semblent d'ores et déjà mis en œuvre, il nous semble que les pré-requis de l'intranet sont bel et bien présents.

II.2.2.2. Aspects politiques, techniques et fonctionnels

Afin de pouvoir utiliser à bon escient ce qui a déjà été mis en place précédemment, il nous faut dès lors percevoir comment les aspects politiques, techniques et fonctionnels ont été traduits. Autrement dit, nous devons retrouver *a posteriori* la phase de préparation de l'intranet à l'œuvre actuellement, pour pouvoir distinguer les contraintes qui ont guidé les développements jusqu'ici et pour les respecter encore. Grâce à cette démarche, nous pourrons percevoir les caractéristiques principales de la situation que nous vivons à l'Institut.

Tout d'abord, au niveau politique, l'ébauche d'intranet manifeste une découpe claire entre les étudiants et les membres du staff. Ces derniers ont accès à l'espace virtuel des premiers, mais l'inverse n'est pas vrai. Par contre, au sein du staff, aucune différence explicite n'apparaît entre le corps professoral, les chercheurs et le personnel administratif. L'infrastructure électronique se base donc sur une bipartite organisationnelle. Par ailleurs, au sein même du fonctionnement des services, le contrôle ultime est toujours laissé à l'une ou l'autre personne responsable de ce service ; ainsi, par exemple, la fonction de réservation de locaux est réalisée par l'envoi d'un courrier électronique au responsable du local, à qui il revient d'accepter ou de refuser cette demande de réservation. Il en va de même pour la réservation de matériel. Ces modalités techniques nous montrent que, jusqu'ici, l'Institut n'est pas entré dans une dynamique de distribution de la prise de décision, telle que Steven Telleen nous l'avait présentée à la fin du chapitre précédent. Cette rigidité organisationnelle tient dans l'essence même de la situation concrète : une faculté au sein d'une université ne peut être assimilée à une entreprise. L'organisation du travail, dans le chef de l'université belge, est principalement structurée en unités séparées, dans lesquelles une hiérarchie importante subsiste : les décisions se prennent surtout en comités, formés conformément au statut de chacun. Par ailleurs, le travail des membres d'une université, qu'ils soient étudiants, chercheurs, professeurs ou administratifs, est très souvent individuel. Au sein de cette structure relativement rigide et individualiste, la communication électronique doit d'abord et avant tout permettre d'améliorer la qualité de la distribution de l'information.

Au niveau technique, nous avons déjà souligné le caractère uniforme de la solution choisie. Opter pour un CGI écrit en scripts Unix revient à garantir une simplicité d'implantation et de maintenance, en même temps qu'une efficacité d'exécution. Les scripts Unix, en effet, ne requièrent pas de compétence excessive : un informaticien n'éprouvera, en principe, aucune difficulté à en comprendre la syntaxe pour, par exemple, les modifier ou les corriger. De plus, des solutions techniquement plus pointues, comme par exemple des applets Java, nécessitent davantage de ressources et prennent donc plus de temps à s'exécuter ! Cette prépondérance pour la simplicité et l'efficacité a également guidé la gestion de l'annuaire : vu le peu de membres de l'Institut, introduire une technologie spécifique apparaîtrait vraisemblablement comme une solution disproportionnée. L'utilisation de fichiers texte semble suffire. En optant pour ces technologies simples et efficaces, l'Institut respecte également les contraintes financières auxquelles toute institution universitaire belge est désormais soumise. L'Institut ne peut octroyer à l'intranet un quelconque budget, ni pour son implantation, ni pour sa maintenance. Employer les ressources déjà disponibles s'impose, dès lors, comme une nécessité absolue : les serveurs, les systèmes d'exploitation, les protocoles… Quant à la maintenance, nous ne pouvons compter sur un *IT Staff* qui y serait consacré à temps plein, comme le suggérait la théorie. Ce sont les membres de l'Institut qui doivent prendre en main cette gestion continue de l'*Institut Wide Web*, en plus de leurs tâches habituelles.

Quant à l'aspect fonctionnel de l'intranet déjà mis en place, il nous apparaît comme très diversifié. Pour les étudiants, des informations très hétéroclites sont rassemblées sur leur espace virtuel ; aucun lien ne semble réunir ces renseignements. Ce manque de structure risque, avec l'accroissement de l'intranet, de perturber les utilisateurs et finalement de les détourner de la mine d'informations que doit constituer le site interne. De plus, aucune application *on-line* n'est encore disponible pour les étudiants. La situation est quelque peu différente pour le staff. Les informations sont plus ciblées ; et les applications à disposition sont bien mises en évidence. Vraisemblablement, un effort de structure et d'intégration a été ici mis en œuvre. Par ailleurs, nous devons également signaler que le site Web de l'Institut propose, en dehors de tout intranet, la consultation des pages personnelles des membres du staff et des étudiants. Tous les étudiants et tous les membres du staff n'ont pas nécessairement de *home page* ; tous les professeurs, quant à eux, en possèdent une (parfois élémentaire). Certaines pages personnelles regorgent d'informations importantes sur les cours dispensés (résumés, sommaires, notes…) L'aspect fonctionnel de l'infrastructure électronique existante nous apparaît donc, finalement, comme peu structurée, voire en construction : il faut un minimum de matière pour voir comment l'agencer.

II.2.3. Prolongements

Au terme de cette analyse de l'existant, quelques conclusions peuvent être tirées. Tout d'abord, il apparaît que l'intranet qui existe déjà à l'Institut peut nous servir de base pour un développement plus étendu : les pré-requis nécessaires à la mise sur pied d'un intranet ont bien été établis. Ensuite, les contraintes qui ont guidé jusqu'ici les aspects organisationnels, techniques et fonctionnels ont été dégagées : nous savons désormais quelles sont les limites inhérentes au projet du Web interne.

Dès lors, nous pouvons nous pencher sur les besoins communicationnels non encore satisfaits à l'Institut pour les intégrer dans le projet intranet qui nous occupe, et qui occupera d'autres encore après nous. C'est grâce à des entretiens personnels que nous allons maintenant

étudier les besoins observés à l'Institut. Pour chacun d'entre eux, nous tenterons d'apporter une description précise (point de vue fonctionnel), et de souligner les contraintes techniques et organisationnelles déjà discernables. Nous pourrons alors, dans un second temps, évaluer ces besoins et les classer selon leur importance et leur priorité, au niveau communicationnel. Les sept questions d'évaluation continue, que nous avons exposées précédemment, guideront implicitement notre recherche des besoins.

Avant toutefois de nous lancer dans cette tâche, nous tenons à relever encore une contrainte de notre travail. Nous nous situons dans le cadre d'un mémoire de Licence. Ce contexte nous impose une échéance stricte. Autrement dit, notre but ne consiste pas à mettre en place un intranet complet : ce serait totalement contraire, d'ailleurs, à la philosophie de l'intranet. Tout au plus, pourrons-nous poser les bases d'un intranet en continuelle progression, agencer les premières briques d'un édifice que d'autres se chargeront de bâtir. C'est pourquoi cette description des besoins doit être considérée comme un point de départ, limité.

II.3. Un relevé des besoins

Afin de structurer les résultats des entretiens que nous avons pu obtenir, nous distinguons les besoins selon leur domaine d'application : les besoins administratifs, pédagogiques et scientifiques. Cette découpe peut parfois paraître artificielle : certains besoins peuvent ressortir de plusieurs catégories et d'autres n'appartenir spécifiquement à aucune… C'est sans doute pourquoi la catégorie des besoins administratifs est la plus fournie : lorsque nous hésitions entre plusieurs possibilités, c'est l'administration qui semblait s'offrir comme le secteur le plus ouvert.

II.3.1. Les besoins administratifs

II.3.1.1. Le trombinoscope

Description du besoin

Dans le jargon de l'Institut, le « trombino » est cette liste des étudiants, où chacun est décrit brièvement et représenté par une photographie, que les membres du staff reçoivent en début d'année. Etablir cette liste et la distribuer, manuellement, requiert un effort important, d'autant plus que la charge administrative est déjà lourde en début d'année académique.

Le problème de la distribution pourrait être réglé par une publication électronique au sein de l'intranet. Seuls les membres du staff pourraient y accéder. Cette diffusion électronique permettrait d'économiser du temps de travail et du papier. Peut-être pourrait-on aussi envisager une conception partiellement automatisée du trombino. Ajouter un élève à une liste reviendrait à compléter un formulaire et à le valider. Il faudrait aussi prévoir des procédures de mise à jour (suppression, modification…).

Contraintes techniques

Le trombino ne peut être publié sur l'intranet que s'il possède le format HTML. Dès lors, il importe que toutes les données soient introduites, soit via un script qui créerait le document HTML, soit via un traitement de texte qui possède un convertisseur automatique en HTML (la version de MS-Word97 de l'Institut possède ce type de

convertisseur). Ce document HTML risque d'être long, et peu maniable pour des suppressions ou des ajouts de données. Il serait également possible de constituer une base de données indépendante, qu'on pourrait consulter via des formulaires déposés sur l'intranet.

Par ailleurs la numérisation des photographies constitue un problème à part entière. Il faudra scanner chaque photo une à une et les intégrer à la bonne place dans le document ou dans la base de données. L'ensemble des photographies risque, de plus, d'occuper une place mémoire très importante sur le serveur. Peut-être le trombino pourrait-il être jumelé avec les pages Web personnelles des étudiants.

Contraintes organisationnelles

Le trombino ne doit être accessible qu'aux membres du staff.

Il faut nommer un responsable de la gestion du trombino qui effectuerait les mises à jour au long de l'année (si un élève abandonne, par exemple, il conviendrait de le retirer de la liste).

Conclusion

Inclure le trombinoscope requiert une étude technique particulière.

II.3.1.2. Le serveur de fax

Description du besoin

L'administration a souvent besoin d'envoyer des fax. Or, à l'Institut, il existe un fax commun que les membres du staff peuvent employer ; quelques-uns possèdent aussi un modem connecté à leur ordinateur qui leur permet d'envoyer des fax.

Contraintes techniques

Implanter un serveur de fax via un intranet semble particulièrement complexe. Par contre, il apparaît comme très facile d'installer un fax partagé pour l'ensemble des membres du staff. Ces derniers pourraient l'utiliser comme un simple pilote d'imprimante.

Contraintes organisationelles

Seuls les membres du staff pourraient bénéficier de ce service.

Conclusion :

Ce besoin, même s'il relève de la communication d'information, ne concerne pas à proprement parler l'intranet. Il pourrait être satisfait plus aisément en dehors de celui-ci.

II.3.1.3. L'agenda partagé

Description du besoin

Pour les services administratifs, contacter un professeur ou un chercheur paraît parfois très compliqué : les cours à dispenser, les colloques, les conférences… viennent sans cesse perturber l'horaire habituel. Pour contacter plus facilement chacun, il faudrait une application d'agenda partagé, telle qu'on peut en trouver sur certains *groupware* (Lotus Notes par exemple). Chacun, hebdomadairement, indiquerait ses heures de

présence et de disponibilité à l'Institut, et éventuellement le lieu où on peut le contacter en cas d'absence. Des formulaires adaptés pourraient permettre une saisie interactive de ces données. Un tableau serait automatiquement généré et serait consultable sur l'intranet. Un courrier électronique automatique pourrait rappeler à chacun de compléter ce formulaire en début de semaine, si cela n'a pas été effectué.

Contraintes techniques

Techniquement, cette fonction nécessite des ressources importantes : un tableau doit être prévu pour chacun des membres du staff et pour chacune des semaines de l'année académique ! Le nombre de pages HTML générées est dès lors très élevé.

Contraintes organisationnelles

Demander aux membres du staff de partager leur agenda semble en fait impossible du point de vue organisationnel. L'agenda des membres du corps professoral, notamment, apparaît bien trop fluctuant pour être prévu hebdomadairement. Tout au plus, serait-il envisageable de déposer sur l'intranet les moments préférés de chaque professeur et assistant pour recevoir les étudiants (une heure de permanence hebdomadaire, en quelque sorte).

Conclusion

Ce besoin ne peut être inclus sur l'intranet. D'ailleurs, il n'a été une réussité dans aucune organisation non bureaucratique. Une information sur les heures de permanence pourrait peut-être le remplacer.

II.3.1.4. Le service « emploi »

Description du besoin

L'intranet pourrait permettre la diffusion des offres d'emploi au sein des étudiants comme au sein du staff. Ce service comporterait deux aspects. D'une part, les Facultés pourraient, sur ce pan de l'intranet, publier les offres d'emploi internes : les postes d'assistant, de chercheur, de professeur qui peuvent se dégager. D'autre part, les entreprises pourraient, elles aussi, déposer des appels de candidature via cette entrée du réseau interne. Un formulaire *ad hoc* pourrait être déposé sur le site public de l'Institut : tout qui voudrait publier une annonce compléterait ce formulaire. Les étudiants comme les membres du staff pourraient alors profiter, *on-line*, de ces informations sur les opportunités de carrières académiques ou privées.

Contraintes techniques

Techniquement, la mise en place de ce service reviendrait à celle d'un panneau d'affichage électronique (valves). Les fonctions d'insertion d'informations et de suppression devraient pouvoir être implémentées. Pour automatiser le plus possible la gestion de ce panneau électronique, il faudrait prévoir des fonctions de mises à jour automatiques (une information vieille de plus de x semaines serait automatiquement supprimée). Par ailleurs, il semble délicat d'ouvrir cet espace aux entreprises : actuellement, rappelons-le, la sécurité de l'intranet est assurée sur un tri des numéros IP des machines, ce qui exclu de l'intranet toute machine extérieure aux Facultés.

Une autre solution technique serait envisageable : un lien hypertexte déposé sur la première page du site Web de l'Institut (accessible par tout internaute) pourrait référencer l'ensemble des étudiants de dernière année (ou tout du moins ceux qui

acceptent d'être insérés dans cette liste) : un simple clic de souris sur ce lien permettrait d'envoyer un courrier électronique à ces derniers. Il suffirait de créer un alias vers tous les étudiants de deuxième licence et de troisième maîtrise qui acceptent d'être contactés par les entreprises et d'inclure cet alias avec le code « mailto : » au sein d'une page HTML. Un autre lien pourrait référencer l'ensemble des membres du staff, selon la même technique. Cet ensemble de deux liens permettrait à n'importe quelle institution (privée ou académique) de contacter tous les étudiants et tous les membres du staff pour leur transmettre de l'information. Cette solution présente néanmoins un risque : une offre d'emploi trop longue (agrémentée de dessins ou de fichiers attachés) peut encombrer le serveur de mail et même le saturer.

Contraintes organisationnelles

Selon la solution technique choisie, les contraintes organisationnelles sont différentes. La gestion d'un panneau d'affichage électronique nécessite la nomination d'un responsable (*Webmaster*) de ce service. La deuxième solution ne nécessite pas de responsable attitré.

Conclusion

Pour un service superficiel à l'administration, le choix de la première solution technique semble d'ores et déjà démesuré. Par contre la deuxième solution rencontre moins bien le besoin exprimé (on ne gère plus un espace d'affichage virtuel, à l'image de ce qui se fait actuellement), mais semble plus réaliste tant au niveau technique qu'au niveau organisationnel.

II.3.1.5. Les inscriptions (examens, cours à option, autres sections)

Description du besoin

Le secrétariat administratif s'occupe de recevoir les diverses inscriptions des étudiants aux obligations académiques. Ainsi, en début d'année, tous les étudiants choisissent leurs cours à option et les transmettent au secrétariat à la faveur d'un formulaire *ad hoc* ; certaines catégories d'étudiants, comme ceux qui effectuent un D.E.A. s'inscrivent à tous les cours qu'ils désirent suivre. A l'approche de la fin de chaque semestre, c'est aux examens qu'il s'agit de s'inscrire. Toutes ces inscriptions pourraient être supportées par l'intranet. Chacune des sections (Licence, Maîtrise, DEA, DGTIC…) disposerait, *on-line*, des formulaires d'inscription correspondant à leurs obligations, selon l'époque académique en cours.

Contraintes techniques

Pour concrétiser l'inscription électronique, deux solutions techniques sont envisageables : soit une solution entièrement automatisée qui enregistre les inscriptions et les insère dans une base de données (fut-elle élémentaire via des fichiers texte), soit une inscription qui passe par le courrier électronique. La première possibilité nécessite une programmation complexe, avec une vérification de toutes les contraintes auxquelles les étudiants sont soumis selon leur programme, le type de cours concerné… La deuxième solution permet de garder un contrôle plus strict sur chacune des opérations, et requiert bien sûr beaucoup moins de programmation : en cas de problème sur l'inscription, il suffit de retourner un mail à l'expéditeur via la fonction « *Reply* » du gestionnaire de courrier. Cette deuxième solution instaure également un dispositif de sécurité interne : une personne qui s'inscrit à un cours ou à

un examen recevrait automatiquement un mail personnalisé de confirmation, ce qui empêcherait toute plaisanterie de mauvais goût (c'est-à-dire se faire passer pour quelqu'un d'autre).

Contraintes organisationnelles

Pour que ce service soit efficace, une attention soutenue devra être accordée à sa maintenance. Les bons formulaires (c'est-à-dire correctement mis à jour) devront être disponibles au bon moment, et pour les bonnes personnes. Un gestionnaire des formulaires d'inscription devra donc nécessairement être nommé : cette tâche est particulièrement lourde puisqu'il s'agit de récolter toutes les informations sur les inscriptions et de sans cesse veiller à leur traduction correcte sur le Web interne.

Conclusion

Ce service peut permettre d'économiser beaucoup de papier. Mais il n'est réalisable qu'à la faveur d'une solide volonté organisationnelle.

II.3.1.6. Une base de données des membres de l'Institut

Description du besoin

Une base de données Microsoft Access, reprenant les adresses des membres de l'Institut, existe au sein du secrétariat des Unités d'Enseignement et de Recherche. Cette base de données pourrait être partagée via l'intranet, de manière à imprimer des étiquettes pour des enveloppes directement.

Contraintes techniques

Permettre un accès à une base de données Access via une page HTML est techniquement possible. Plusieurs outils, issus du même constructeur, peuvent être employés : on pense particulièrement à ASP (*Active Page Server*). Toutefois, cette technique est propriétaire ; elle nécessite un système d'exploitation 32 bits (de la gamme Windows) tant pour le serveur que pour le poste client. De plus, il faut acheter le logiciel de développement ASP.

Contraintes organisationnelles

La base de données dont il est question ici appartient plutôt aux ressources du secrétariat des UER. Mettre cette ressource en partage, via l'intranet, entraîne une certaine perte de contrôle pour ce secrétariat. Cette perte de contrôle n'est pas nulle : l'impression des étiquettes est souvent requise pour des événements relativement importants à l'Institut. Jusqu'ici, le secrétariat des UER, grâce à cette base de données et à l'impression d'étiquettes, est nécessairement au courant de ce type de manifestation. Finalement, c'est un statut qui est en jeu : celui d'une centrale d'information. Des tensions d'ordre politique accompagneraient donc la mise en place ce service.

Conclusion

Ce besoin requiert : une technologie propriétaire pour son développement technique sur l'intranet, l'investissement dans cette technologie, une vigilance accrue du risque politique entre secrétariats administratifs. En vaut-il vraiment la peine ?

II.3.1.7. Les comptes-rendus des réunions et des prises de décision

Description du besoin

De nombreuses réunions se déroulent au sein des murs de l'Institut (la Cocon, le Conseil, le Bureau). Les procès-verbaux de chacune de ces réunions ne sont distribués qu'aux personnes qui y ont assistées. Les décisions prises lors de ces réunions sont, *de jure*, accessibles à tout un chacun.

Actuellement, les prises de décision sont consignées dans des documents, déposés dans des répertoires partagés du réseau : chacun, par l'intermédiaire du réseau, et non via l'intranet, peut déjà y accéder. Quant aux PV, ils sont d'ores et déjà envoyés par mail à tous les participants de la réunion en question.

Toutes ces informations pourraient être déposées sur l'intranet, de manière à être consultée *on-line* par qui de droit.

Contraintes techniques

Le problème technique consiste essentiellement ici dans la sécurité interne de l'intranet. Il est absolument impossible, dans l'état actuel des choses, de distinguer les membres de tel comité, par rapport à ceux de tel autre, puisque seuls les membres du staff peuvent être différenciés des étudiants. Etablir un filtre pour les procès-verbaux des réunions nécessite dès lors la création d'une procédure spéciale (l'accès via un mot de passe défini, par exemple).

Contraintes organisationnelles

En ce qui concerne les PV de réunion, il est important de continuer à les envoyer par courrier électronique aux membres de celle-ci. L'intranet, en effet, ne doit pas entraîner un mécontentement des utilisateurs.

Quant aux relevés des prises de décision, l'intranet pourrait effectivement en permettre une distribution plus conviviale.

Conclusion

Le dépôt des PV de réunion sur l'intranet apparaît comme techniquement lourd à réaliser et redondant par rapport à la procédure actuelle (envoi d'un mail). Le dépôt des relevés de prises de décision apparaît plus réalisable.

II.3.1.8. La gestion des indisponibilités des professeurs

Description du besoin

Le secrétariat des étudiants compte deux activités sur l'année académique pour lesquelles la communication avec le corps enseignant est cruciale : l'établissement des horaires de cours au mois de septembre et celui des horaires d'examens, en janvier et en juin. L'intranet pourrait permettre de récolter plus facilement les indisponibilités de chacun des professeurs aux moments clefs de l'année et ainsi d'établir les horaires plus efficacement. Ce service n'est pas identique à l'agenda partagé, que nous avons vu précédemment. Ici, nous nous situons à des moments exceptionnels, où l'horaire hebdomadaire doit être modelé ou modifié. Même si l'agenda partagé était mis en œuvre, on ne pourrait faire l'économie des procédures particulières consacrées aux débuts d'année et aux examens.

Contraintes techniques

Techniquement, de simples tableaux à compléter *on-line* sur l'intranet pourraient faire l'affaire.

Contraintes organisationnelles

Ce service permettrait bien sûr au secrétariat une économie de temps de travail. Les professeurs, en effet, ne pourraient indiquer que des moments libres sur des plages horaires que chacun viendrait compléter. Seulement, on peut craindre que la philosophie du « premier arrivé, premier servi » ne convienne pas aux professeurs qui sont justement les plus occupés, et dont la priorité n'est pas de communiquer leurs indisponibilités. Par ailleurs, la consultation de l'intranet et le passage par une application spécifique risque de prendre plus de temps qu'un simple coup de téléphone.

Conclusion

Ce service risque de ne pas être effectivement utilisé par tous du fait des contraintes organisationnelles. Une utilisation par quelques-uns suffit néanmoins à améliorer la communication ; d'autant plus qu'une fois introduite, cette fonctionnalité pourra convaincre petit à petit.

II.3.1.9. Les commandes à l'économat

Description du besoin

Pour effectuer des commandes de matériel à l'économat, il existe aujourd'hui une procédure manuelle assez lourde. Cette dernière pourrait être remplacée avantageusement par un formulaire électronique simplifié.

Contraintes techniques

Un formulaire HTML qui débouche sur l'envoi d'un mail à l'économe permettrait de satisfaire l'utilisateur. Pour construire ce formulaire, deux solutions peuvent s'envisager : soit le développement d'une application spécifique, soit l'utilisation d'un traitement de texte et d'un convertisseur HTML. Une solution plus pointue viserait la gestion de base de données…

Contraintes organisationnelles

Sauf contre-indication de l'économe, aucune contrainte organisationnelle ne pèse sur ce besoin.

Conclusion

C'est un service rapidement implémentable.

II.3.1.10. La liste des étudiants pour les entreprises

Description du besoin

Chaque étudiant peut autoriser l'Institut à distribuer ses coordonnées aux entreprises qui le souhaitent. Ce processus s'effectue aujourd'hui de manière entièrement manuelle : chacun doit compléter un formulaire et le signer. Le secrétariat des

étudiants de l'Institut établit ensuite la liste et l'envoie au secrétariat central des Facultés qui gère les contacts avec l'extérieur.

Au lieu d'utiliser des formulaires papier, l'intranet pourrait offrir des formulaires électroniques. La liste des étudiants pourrait alors être élaborée automatiquement. On pourrait même imaginer que cette liste soit consultable de l'extérieur par les entreprises et soit combinée avec le service « emploi ». Ainsi, l'alias de contact des $2^{\text{èmes}}$ licences et $3^{\text{èmes}}$ maîtrises ne reprendrait que les membres de cette liste.

Contraintes techniques

Les difficultés techniques inhérentes à ce service relèvent plutôt de la sécurité interne : comment être sûr de l'identité de celui qui autorise la publication de coordonnées vers les entreprises, puisqu'elle tient dans le numéro IP de la machine utilisée ? Tout comme nous l'avons proposé pour les inscriptions électroniques, l'envoi d'un mail confirmatif personnalisé pourrait permettre de déjouer les fraudes : si quelqu'un se fait passer pour un autre, cet autre en sera automatiquement averti et pourra annuler la procédure.

Contraintes organisationnelles

L'élaboration de la liste des étudiants ne pose aucun problème organisationnel. Une fois celle-ci constituée, le secrétariat des étudiants pourra l'envoyer au secrétariat central qui prendra le relais. Par contre, la publication de cette liste sur le Web, de manière à ce que les entreprises puissent la consulter *on-line*, revient à priver le secrétariat central des FUNDP d'un pouvoir qui lui appartient. Un examen plus poussé des tenants et aboutissants organisationnels est donc à prescrire ici.

Conclusion

Cette fonctionnalité doit requérir de la prudence tant au niveau technique (sécurité interne) qu'au niveau organisationnel (relations politiques entre le secrétariat central des FUNDP et le secrétariat des étudiants de l'Institut).

II.3.1.11. Les tractations avec les agences de voyage

Description du besoin

Certains professeurs passent par le secrétariat des UER pour contacter une agence de voyage et régler les détails de déplacements à l'étranger en vue de colloques ou de conférences. L'Institut travaille toujours avec les deux mêmes agences. Puisque celles-ci disposent d'une adresse électronique, l'intranet faciliterait les contacts par l'intermédiaire d'un formulaire convivial. Ce dernier permettrait, notamment, un échange plus direct des diverses potentialités de l'agence quant au voyage recherché.

Contraintes techniques

Il suffit de connaître l'adresse électronique de ces agences.

Contraintes organisationnelles

Si les professeurs passent par les secrétariats pour régler les détails de voyage à l'étranger, c'est bien pour en être débarrassés eux-mêmes. Dès lors, il est illusoire de croire qu'ils vont employer le formulaire déposé sur l'intranet. Ce formulaire pourrait tout au plus être utilisé par le secrétariat.

Conclusion

Techniquement simple, ce service ne peut atteindre le but initialement fixé, compte tenu de la situation organisationnelle. D'autant plus qu'un coup de téléphone est, dans ce cas, sans doute plus efficace qu'un courrier électronique.

II.3.2. Les besoins pédagogiques

II.3.2.1. L'horaire des cours en temps réel

Description du besoin

En marge de l'horaire officiel des cours, se décident, jour après jour, des déplacements de cours, des récupérations, des séances de rattrapage, des conférences… L'horaire officiel est donc agrémenté de maintes modifications ponctuelles, que rassemble le secrétariat des étudiants et qui constituent autant d'avis aux valves. Il est très difficile, pour les étudiants, pour les professeurs comme pour les secrétariats, d'obtenir une vision unifiée de la situation. L'intranet pourrait, non seulement diffuser l'horaire officiel, mais aussi un horaire « en temps réel », mis à jour continuellement, que chacun pourrait consulter. Cet horaire prendrait la forme d'un tableau hebdomadaire, qui serait complété selon les changements demandés au secrétariat des étudiants.

Contraintes techniques

A priori, la mise en place d'un tel outil serait facilement réalisable techniquement. Les changements d'horaire pourraient être communiqués au secrétariat des étudiants par mail ou par téléphone, en fonction des disponibilités décrites par le tableau consultable *on-line*. Ce secrétariat pourrait alors utiliser un tableau Excel ou Word puis un convertisseur HTML pour produire, ou pour mettre à jour, l'horaire en temps réel. Le déposer à l'endroit *ad hoc* sur le serveur ne doit pas poser de problèmes particuliers. Cette suite chronologique de tableau pourrait être accessible par un simple lien hypertexte via l'intranet.

Contraintes organisationnelles

Pour que ce service soit effectivement opérationnel, il faut que tous les changements d'horaire passent par le secrétariat des étudiants, ce qui est le cas actuellement.

La consultation de ces horaires en temps réel nécessite l'utilisation d'un ordinateur (appartenant au parc informatique des Facultés, puisque la sécurité de l'intranet l'impose). Cette consultation électronique risque de prendre plus de temps qu'une consultation sur un panneau d'affichage : un ou plusieurs ordinateurs dédicacés à cette seule fonction et facilement accessibles peuvent combler ce manque à gagner.

Conclusion

C'est un service qui ne requiert pas de ressources techniques excessives, qui ne supporte pas de contraintes organisationnelles importantes et qui serait très utile pour tous.

II.3.2.2. La publication du programme analytique des cours

Description du besoin

Pour les étudiants (voire les futurs étudiants), le programme analytique des cours, qui détaille chacun des cours dispensés à l’Institut pour chacune des années et des sections, constitue une mine de renseignements. Publier ces informations sur l’intranet permettrait de rassembler toutes les données concernant les cours en un endroit virtuel bien spécifique.

Contraintes techniques

Le document actuel existe déjà en Word97. La conversion en HTML est immédiate.

Prévoir une consultation pour des futurs étudiants (c'est-à-dire extérieurs à l’Institut) est impossible selon la procédure de sécurité actuelle.

Contraintes organisationnelles

Il serait très intéressant de concentrer toutes les informations sur les cours, soit sur le programme analytique, soit à un endroit sur l’intranet proche de ce programme. On pense surtout aux nombreux renseignements qui émaillent les pages personnelles de certains professeurs.

Conclusion

C’est une information importante dont la publication ne demande (presque) aucun effort. De plus, ce peut être un point de départ pour la centralisation des informations sur les cours au niveau de l’intranet.

II.3.2.3. La publication du guide de l’étudiant

Description du besoin

En début de chaque année académique, un guide de l’étudiant est distribué aux nouveaux arrivants. Ces guides sont souvent égarés rapidement. En publiant ceux-ci sur l’intranet, on permettrait à chacun de retrouver l’information.

Contraintes techniques

Ce document doit simplement être converti en HTML. Remarquons toutefois qu’il est constitué d’une trentaine de pages… la place occupée n’est donc pas nulle.

Contraintes organisationnelles

Comme le contenu de ce guide ne change au maximum qu’une fois par an, sa publication électronique n’est absolument pas contraignante.

La publication électronique peut-elle remplacer la version papier ? Autrement dit, peut-on demander aux jeunes arrivants à l’Institut de surfer les premiers jours de leur arrivée sur le Net ? Dans l’état actuel des choses, il nous semble qu’il faut laisser les versions papier et électronique coexister.

Conclusion

C’est un service facilement implémentable, mais dont l’utilité reste très ponctuelle.

II.3.2.4. Les propositions et choix de mémoire

Description du besoin

Dès la première licence ou la deuxième maîtrise, selon le curriculum emprunté, les étudiants sont invités à choisir un sujet de mémoire parmi une liste proposée par les divers professeurs. Actuellement, cette liste est publiée aux valves.

L'intranet pourrait rendre plusieurs services autour de cette situation. Tout d'abord, il pourrait permettre une publication et une diffusion électronique de la liste des sujets ; ainsi, lorsqu'un professeur découvre un nouveau sujet, il suffit simplement de mettre cette liste à jour. Cette liste est déjà disponible actuellement via le réseau classique. Ensuite, une application spécifique pourrait permettre à chaque étudiant de sélectionner le(s) sujet(s) qu'il préférerait aborder. Enfin, la liste des étudiants avec les mémoires qui leur ont été attribués pourrait également figurer sur l'intranet.

Contraintes techniques

La diffusion de liste s'apparente à une simple conversion HTML de documents produits grâce au traitement de texte.

La sélection et le choix de sujets de mémoire par les étudiants pourrait s'effectuer via un formulaire spécifique ; le nombre de choix par sujet pourrait même s'afficher au et à mesure sur liste.

Un problème de sécurité interne subsiste… Il est impossible d'être sûr de l'identité de celui qui envoie le formulaire. De nouveau ici, un système de confirmation automatique par mail envoyé à l'étudiant permettrait de déjouer les malversations.

Contraintes organisationnelles

Les propositions et les choix de mémoire ne constituent pas des activités très formelles à l'Institut. Souvent, des compromis s'instaurent entre professeurs et étudiants quant aux sujets proposés. Il est donc utile de vouloir améliorer la communication, mais il serait illusoire de vouloir tout programmer.

Conclusion

Cet ensemble de fonctionnalités autour du mémoire a certes une portée limitée, mais il est aisément réalisable et ne rencontre aucune contre-indication sur le plan politique.

II.3.2.5. Les valves électroniques

Description du besoin

Les valves, telles que nous les connaissons aujourd'hui, désignent ces tableaux d'affichage qui constituent un canal de communication privilégié entre les divers acteurs de l'Institut d'Informatique : les étudiants, les secrétariats et le personnel scientifique et académique. Les valves sont différentes selon les personnes auxquelles elles s'adressent (staff ou étudiants), et selon leur caractère officiel ou officieux (on trouve des valves officieuses avec des offres d'emploi pour les étudiants, par exemple). Toutes les valves sont composées d'avis divers, dont la longueur dépasse rarement une page.

Puisqu'un intranet est un outil élaboré de communication au sein d'une organisation, la migration électronique des valves semble tout indiquée. Ainsi, on peut imaginer des « valves électroniques », composées de pages Web reprenant une suite d'avis. Il serait particulièrement opportun que ces valves électroniques puissent être auto-nettoyantes :

lorsqu'un avis aurait dépassé une date de validité, il serait automatiquement effacé. Dans la foulée de cette gestion automatique, on pourrait également concevoir un système de publication d'avis à l'avance : un avis pourrait être déposé sur les valves pour une date ultérieure.

Contraintes techniques

Techniquement, les valves électroniques pourraient consister en une gestion de fichiers de format texte. Ajouter un avis, en supprimer… reviendrait à effectuer le traitement correspondant sur le fichier. La mise à jour automatique pourrait être réalisée par l'exécution régulière d'un programme adapté. Il faudra néanmoins veiller à obtenir une certaine efficacité lors d'une exécution en situation réelle (c'est-à-dire avec de nombreux avis qui cohabitent).

De manière plus pointue, une gestion de base de données comme telle permettrait aussi de mettre sur pied ce service.

La sécurité des valves pourra être simplement assurée par une distribution astucieuse des diverses fonctions au sein des membres de l'Institut : ajout, suppression et consultation.

Contraintes organisationnelles

Transformer un moyen de communication aussi important que des panneaux d'affichage nécessite des efforts d'adaptation de la part de chacun des acteurs de l'Institut. Les étudiants, tout d'abord, auront à consulter ces valves électroniques : ils devront quotidiennement, voire deux fois par jour, se connecter à l'intranet et lire la rubrique des valves. Consulter les valves relève d'une obligation stricte, mais la consultation électronique risque de prendre plus de temps qu'un simple coup d'œil en passant devant un panneau. Cet effort d'adaptation concerne aussi les membres du staff qui possèdent leurs propres valves.

Les secrétariats connaîtront vraisemblablement une charge plus importante de travail, car le système actuel des valves « papiers » devra coexister avec le système électronique pendant un certain temps. Cette double charge devra être allégée par un système performant d'autogestion des valves électroniques.

La question de la distribution du pouvoir autour de ces valves se pose aussi : qui aura le droit d'afficher un avis électronique et d'en retirer ? Faudra-t-il limiter ces droits au secrétariat des étudiants, afin de prolonger la situation actuelle ? Ou est-ce l'occasion d'étendre cette possibilité à l'ensemble des membres du staff ?

Malgré une gestion automatisée, un responsable (*Webmaster*) des valves électroniques devra être désigné, afin de veiller à leur bon fonctionnement technique et organisationnel.

Conclusion

Cette fonctionnalité apparaît comme exigeante au niveau technique et ambitieuse sur le plan organisationnel. Cependant, elle est surtout symbolique quant au projet intranet : intégrer ces valves au sein d'un Web interne représente une évolution fondamentale de la communication classique vers les voies électroniques.

II.3.3. Les besoins scientifiques

II.3.3.1. Le « welcome staff pack » électronique

Description du besoin

De nombreuses questions, incompréhensions ou difficultés de jeunes membres du personnel viennent probablement d'un manque d'information. En principe, lorsque quelqu'un est engagé à l'Institut au sein du staff, il reçoit un « welcome staff pack », c'est-à-dire un dossier avec toutes les informations indispensables. Mais un dossier s'égare, se détériore… Si ce dossier d'accueil et d'information était consultable à partir de l'intranet, cette information serait toujours disponible.

Contraintes techniques

Il s'agit d'une simple conversion HTML de documents existant sous format Word.

Contraintes organisationnelles

Cette information ne concerne qu'un tout petit pourcentage des membres de l'Institut, et pour des moments très ponctuels.

Conclusion

C'est un service simple et rapide, mais dont l'intérêt est limité.

II.3.3.2. Une liste des ouvrages et des mémoires disponibles

Description du besoin

A l'Institut, de nombreux mémoires de fin d'étude ont été défendus. Le secrétariat des études possède un relevé de ceux-ci, mais cette information n'est pas distribuée. Pour les ouvrages contenus dans les murs de notre institution, aucune liste n'existe encore ; il faudrait donc d'abord répertorier tous les ouvrages, pour ensuite en communiquer la liste via l'intranet.

Contraintes techniques

Seul un convertisseur HTML est nécessaire pour passer du traitement de texte à la publication de la liste des mémoires sur l'intranet.

Puisque la liste des ouvrages n'est pas encore établie, on peut envisager deux possibilités : un relevé sous format standard ou une base de données comme telle. Tout dépend du nombre d'ouvrages : s'il est important, une base de données permettra une gestion plus adaptée.

Contraintes organisationnelles

Publier la liste des mémoires effectués à l'Institut ne met en jeu aucune contrainte organisationnelle. Etablir la liste des ouvrages disponibles, par contre, nécessite l'accord des membres du staff qui emploient actuellement ces ouvrages : il serait malvenu de les répertorier sans demander l'autorisation de ceux qui les utilisent.

II.3.3.3. La publication des bourses possibles

Description du besoin

Les bourses d'étude et de recherche sont aussi précieuses que méconnues. Les secrétariats ont connaissance de quelques bourses, mais ne peuvent pas toujours mettre cette information à disposition de ceux qui la cherchent. Publier sur l'intranet les possibilités de bourses et leurs modalités de candidature serait la solution.

Contraintes techniques

Il suffirait de construire le document HTML.

Contraintes organisationnelles

Ce service requiert une certaine maintenance. Les conditions d'octroi des bourses changent annuellement ; il convient donc de nommer un responsable de l'espace qui accueillerait cette information sur l'intranet. Diffuser un renseignement erroné rendrait en fait un mauvais service !

Conclusion

Cette fonction peut être très utile à condition que quelqu'un accepte de prendre sa maintenance en charge.

II.3.3.4. L'organisation de colloques

Description du besoin

L'Institut organise parfois des colloques. Les aspects pratiques d'une telle manifestation sont nombreux et diversifiés. Nous pouvons les structurer en trois activités principales. Tout d'abord, le colloque doit être annoncé : de nombreuses lettres et de nombreux courriers électroniques sont envoyés en guise de publicité au colloque. Ensuite, les inscriptions qui arrivent par le courrier postal sont dépouillées pour constituer la liste des participants. Enfin, les réservations indispensables au bon déroulement du colloque doivent être effectuées : réservation du parking des Facultés, réservation du restaurant universitaire pour les repas, réservation des salles de conférence et de séminaire…

L'intranet pourrait fournir de l'aide pour chacune de ces étapes de la réalisation d'un colloque : les annonces publicitaires s'effectueraient au moyen d'une procédure automatisée, les inscriptions seraient réalisées au moyen d'un formulaire déposé sur le Web, et les réservations consignées dans un aide-mémoire interactif.

Contraintes techniques

Cette fonction requiert une maîtrise technique relativement avancée. Vraisemblablement, la publicité doit être réalisée grâce à une base de données des personnes à contacter : il faut donc prévoir l'envoi de courrier électronique en combinaison avec des accès au sein de cette base de données. La mise sur pied de formulaire d'inscription au colloque et la réception de cette inscription dans un fichier ne poserait aucune difficulté sérieuse ; de nouveau, on pourrait ici constituer une base de données, ce qui augmenterait la difficulté technique. Prévoir un aide-mémoire convivial des réservations à effectuer, avec éventuellement des liens vers les adresses électroniques des personnes à contacter, ne constitue pas une difficulté technique.

Pour permettre à des extérieurs de s'inscrire électroniquement au colloque, il faut, soit modifier la gestion de la sécurité, soit inclure le formulaire dans un espace accessible par tous (sur le site Web de l'Institut).

Contraintes organisationnelles

Ce service requiert une étude toute particulière sur l'organisation d'un colloque. La troisième partie, à savoir la gestion des réservations, peut même devenir du *workflow*, si on pousse la logique jusqu'au bout. Il conviendra donc, lors de la mise en œuvre de cette fonction, d'observer consciencieusement la succession des tâches et des sous-tâches réalisées par le secrétariat des UER, et d'en dresser le diagramme des flux.

Pour permettre un contrôle centralisé du colloque, on évitera de développer des fonctions trop complexes, qui nécessitent une compétence technique exceptionnelle : mieux vaut privilégier des moyens plus simples, à la portée de tous, qui emploient des outils existants (passer par le courrier électronique, par exemple). Le contrôle de l'organisation du colloque doit, de ce fait, être laissé au secrétariat compétent.

Dès lors, aucun *Webmaster* ne doit être désigné pour ce service.

Conclusion

Ce service semble ambitieux du point de vue technique, mais ne rencontre pas d'opposition au niveau organisationnel. Il requiert une excellente préparation.

II.4. Essai d'évaluation (*scoring*)

Nous avons désormais, grâce aux entretiens individuels que nous avons pu obtenir, une liste de besoins. Bien sûr, tous ne peuvent pas immédiatement être satisfaits. Certains même ne méritent peut-être pas d'être effectivement réalisés au sein de l'intranet. Nous devons donc procéder à une évaluation et à un classement de ces besoins afin de discerner ceux qui revêtent une importance prioritaire.

II.4.1. Choix des critères d'évaluation

Pour effectuer ce classement des besoins, nous devons nous donner des critères d'évaluation, aussi objectifs que possible. Pour établir ces derniers, nous nous inspirerons de la méthode dite « de Benson »[73]. Cette méthode donne quelques critères d'évaluation, avec un ensemble de scores qui leur correspond. Ainsi les facteurs économiques peuvent obtenir une somme de 250 points, alors que les facteurs techniques doivent se contenter d'un maximum de 150 : chaque facteur se décompose en critères plus précis qui possèdent chacun leur score maximum. Une fois ce *scoring* réalisé, il suffit de classer les projets selon leur résultat : les plus importants étant ceux qui atteindront les scores les plus élevés.

Cependant, plutôt que de distinguer les facteurs économiques et techniques, comme le préconise cette méthode, nous utiliserons une distinction plus adéquate au cadre académique que nous connaissons : nous évaluerons d'une part l'apport d'une proposition (en termes organisationnels, car un apport technique entraîne toujours un risque), et d'autre part le risque qui l'accompagne (en termes organisationnels et techniques). Contrairement à Benson, nous

---

[73] Cf. PARKER M. M., BENSON R. J., TRAINOR H. E., *Information Economics-Linking Business Performance to Information Technology*, Prentice Hall, 1988.

ne chercherons pas à obtenir un *scoring* final pour chaque proposition. Soustraire le score des risques encourus au score des avantages perçus, par exemple, nous serait trop restrictif. Nous nous contenterons d'évaluer chaque projet en prenant en compte le score des avantages et celui des risques. Moins formelle, notre méthode laisse davantage la place au jugement personnel et donc permet de mieux tenir compte des particularités de chaque projet.

Pour évaluer l'apport d'un service, nous tenterons de quantifier la plus-value qu'il constitue pour l'intranet en tant que canal de communication ; autrement dit, c'est l'avantage de ce service comme information à distribuer que nous tenterons d'apercevoir. Pour cerner le risque encouru par la réalisation d'un besoin, nous utiliserons les catégories du risque habituelles[74], en omettant toutefois le risque d'affaire et le risque systémique (qui sont davantage liés à une situation commerciale). Alors que la méthode de Benson met plus l'accent sur les facteurs économiques que sur les facteurs techniques, nous insisterons davantage sur les apports plutôt que sur les risques : ainsi, si un besoin se révèle particulièrement utile à réaliser, mais que les risques encourus sont importants, nous estimons qu'il peut valoir la peine d'être concrétisé.

En ayant à l'esprit les contraintes inhérentes à la situation, que nous avions évoquées dans le point 2.2.2. de ce chapitre (essentiellement, le manque de budget et l'inertie d'une organisation académique), nous proposons les critères d'évaluation suivant, avec leur score maximal respectif, rassemblés dans un tableau :

Critères d'évaluation			**Scores max.**		
Apport de la proposition			**500**		
	Utilité administrative			100	
	Utilité pédagogique			100	
	Utilité scientifique			100	
	Valeur de l'information partagée			100	
	Intérêt pour tous			100	
Risque de la proposition			**350**		
	Risque technique			100	
		expérience technique en la matière			*50*
		incertitude des spécifications			*50*
	Risque lié à la taille du projet			50	
		estimation de la taille du projet			*50*
	Risque organisationnel			100	
		degré d'innovation organisationnelle			*50*
		intensité des conflits prévisibles			*50*
	Risque de complexité			100	
		complexité intrinsèque de la tâche			*50*
		connaissance du domaine par les futurs utilisateurs			*50*

Figure 5 : Les critères d'évaluation du scoring

Décrivons brièvement ces critères. Un besoin peut concerner un ou plusieurs secteurs d'activité à l'Institut : nous avons choisi de nous tenir à la distinction classique des secteurs administratif, pédagogique et scientifique, et d'évaluer, pour chacun d'entre eux, le service

[74] Pour l'analyse du risque, cf. LESUISSE R., *Théorie des organisations : gestion de projets informatiques,* Namur, FUNDP, 1999, notes de cours polycopiées.

qu'apporte l'information en question : est-il nécessaire, utile, ou superflu ? Le troisième critère, la valeur de l'information partagée, permet de se concentrer davantage sur l'importance intrinsèque de celle-ci ; autrement dit, il s'agit de se demander si la perte de cette information serait catastrophique, importante ou secondaire. Le dernier critère pour l'apport d'un besoin permet de mesurer la qualité distributive de l'information concernée : le partage de cette information est-il superflu, intéressant ou nécessaire ? Chaque critère de mesure de l'apport de l'information est coté sur un maximum de 100 points.

Pour mesurer le risque encouru par une proposition, nous avons détaillé précisément chaque catégorie de risque. Le risque technique se perçoit par l'inexpérience technique en la matière (s'agit-il d'une technique déjà employée, peu employée, pas du tout employée, totalement novatrice) et par l'incertitude des spécifications (l'utilisateur sait-il précisément ce qu'il veut, ou risque-t-il d'introduire des changements dans ces besoins au cours du développement). Un seul critère suffit pour décrire le risque lié à la taille du projet, à savoir l'estimation de cette dernière. Pour évaluer le risque organisationnel, nous utiliserons le degré d'innovation organisationnelle (l'information en question amène-t-elle, au sein du fonctionnement organisationnel, des transformations importantes, secondaires ou minimes) et l'intensité des conflits prévisibles (à cause de cette information, surgiront des conflits importants, secondaires ou minimes). Enfin, le risque de complexité se ramène à deux questions : la tâche est-elle, en soi, très complexe, moyennement complexe, facile (complexité intrinsèque de la tâche) et les futurs utilisateurs connaissent-ils déjà bien le domaine de l'information concernée, un peu, pas du tout (méconnaissance du domaine par les futurs utilisateurs). Chaque critère d'évaluation du risque est coté sur un maximum de 50 points, ce qui entraîne une diminution de l'importance du risque lié à la taille du projet, puisqu'un seul critère est utilisé.

Un service absolument utile et valable comme élément de l'intranet, et qui n'occasionnerait aucune prise de risque pourrait donc avoir un score d'avantages de 500 points et un score de risques nul. Par contre, un service totalement inutile mais excessivement risqué serait coté à 0 point d'avantages et 350 de risques. Entre ces deux extrêmes, toutes les nuances sont possibles.

II.4.2. *Scoring* des besoins

Désormais, nous avons une liste des besoins et des critères d'évaluation. Nous pouvons donc croiser ces données et effectuer le *scoring* à proprement parler, en tenant compte de la description fonctionnelle et des contraintes techniques et organisationnelles que nous avons dégagées précédemment pour chacun des besoins décrits. Le tableau suivant contient le détail de nos choix personnels d'évaluation[75].

[75] Les scores sont le fruit d'une analyse personnelle. Ils sont donc relatifs, et peuvent prêter au débat, voire à la critique. Mais, ils nous semblent constituer une indication suffisamment enracinée dans l'analyse fonctionnelle, technique et organisationnelle pour déboucher sur une comparaison juste des divers projets.

Page blanche

Besoins	Apports						Risques							
	Utilité administrative	Utilité pédagogique	Utilité scientifique	Valeur de l'info	Intérêt pour tous	**Scoring apports**	Inexpérience technique	Incertitude spécifications	Taille du projet	Innovation organisationnelle	Intensité des conflits	Complexité de la tâche	Méconnaissance par les util.	**Scoring risques**
3.1. Les besoins administratifs														
3.1.1. Le trombinoscope	80	80	0	70	60	**290**	25	0	20	15	5	15	5	**85**
3.1.2. Le serveur de fax	90	0	30	0	60	**180**	10	0	5	10	0	10	10	**45**
3.1.3. L'agenda partagé	70	20	10	40	30	**170**	15	20	40	50	45	35	15	**220**
3.1.4. Le service « emploi »	20	0	0	40	30	**90**	5	10	5	10	0	5	10	**45**
3.1.5. Les inscriptions	80	50	0	80	80	**290**	0	10	20	20	20	20	5	**95**
3.1.6. Une base de données partagée	50	0	10	30	5	**95**	40	25	20	10	35	30	40	**200**
3.1.7. Les PV et prises de décision	5	5	5	60	60	**135**	20	10	10	0	5	20	0	**65**
3.1.8. Les indisponibilités des profs	80	70	0	50	70	**270**	0	5	10	10	15	10	5	**55**
3.1.9. Les commandes à l'économat	50	10	10	10	30	**110**	0	5	5	5	5	5	5	**30**
3.1.10. La liste des étudiants pour les entreprises	50	0	0	30	20	**100**	0	5	5	5	5	5	5	**30**
3.1.11. Les agences de voyage	40	10	20	5	1	**76**	0	15	5	10	20	5	15	**70**
3.2. Les besoins pédagogiques														
3.2.1. L'horaire en temps réel	90	90	0	80	90	**350**	5	0	10	15	10	5	15	**60**
3.2.2. Le programme analytique	30	80	10	80	80	**280**	0	5	10	5	5	5	10	**40**
3.2.3. Le guide de l'étudiant	30	80	0	60	30	**200**	0	0	5	5	5	5	5	**25**
3.2.4. Les propositions et choix de mémoire	70	75	30	80	40	**295**	10	15	20	10	5	20	15	**95**
3.2.5. Les valves électroniques	90	90	90	90	90	**450**	5	10	25	30	25	25	15	**135**
3.3. Les besoins scientifiques														
3.3.1. Le « welcome staff pack »	30	30	70	60	20	**210**	0	5	5	5	5	5	5	**30**
3.3.2. Une liste des ouvrages et des mémoires	5	60	60	80	80	**285**	10	20	15	5	10	15	10	**85**
3.3.3. Les bourses possibles	20	20	70	70	60	**240**	0	10	10	5	0	15	5	**45**
3.3.4. Les colloques	80	80	80	80	40	**360**	25	20	30	15	10	30	15	**145**

Figure 6 : Tableau complet du scoring des besoins

II.4.3. Structuration des besoins

Pour établir une structuration des besoins, selon leur degré d'importance et leur niveau de risque, nous avons rassemblé dans le tableau suivant nos données finales.

Besoins	Apports	Risques
3.1. Les besoins administratifs		
3.1.1. Le trombinoscope	290	85
3.1.2. Le serveur de fax	180	45
3.1.3. L'agenda partagé	170	220
3.1.4. Le service « emploi »	90	45
3.1.5. Les inscriptions	290	95
3.1.6. Une base de données partagée	95	200
3.1.7. Les PV et prises de décision	135	65
3.1.8. Les indisponibilités des profs	270	55
3.1.9. Les commandes à l'économat	110	30
3.1.10. La liste des étudiants pour les entreprises	100	30
3.1.11. Les agences de voyage	76	70
3.2. Les besoins pédagogiques		
3.2.1. L'horaire en temps réel	350	60
3.2.2. Le programme analytique	280	40
3.2.3. Le guide de l'étudiant	200	25
3.2.4. Les propositions et choix de mémoire	295	95
3.2.5. Les valves électroniques	450	135
3.3. Les besoins scientifiques		
3.3.1. Le « welcome staff pack »	210	30
3.3.2. Une liste des ouvrages et des mémoires	285	85
3.3.3. Les bourses possibles	240	45
3.3.4. Les colloques	360	145

Figure 7 : Tableau synthétique du scoring des besoins

Les graphique suivant illustre visuellement les poids respectifs des apports et des risques.

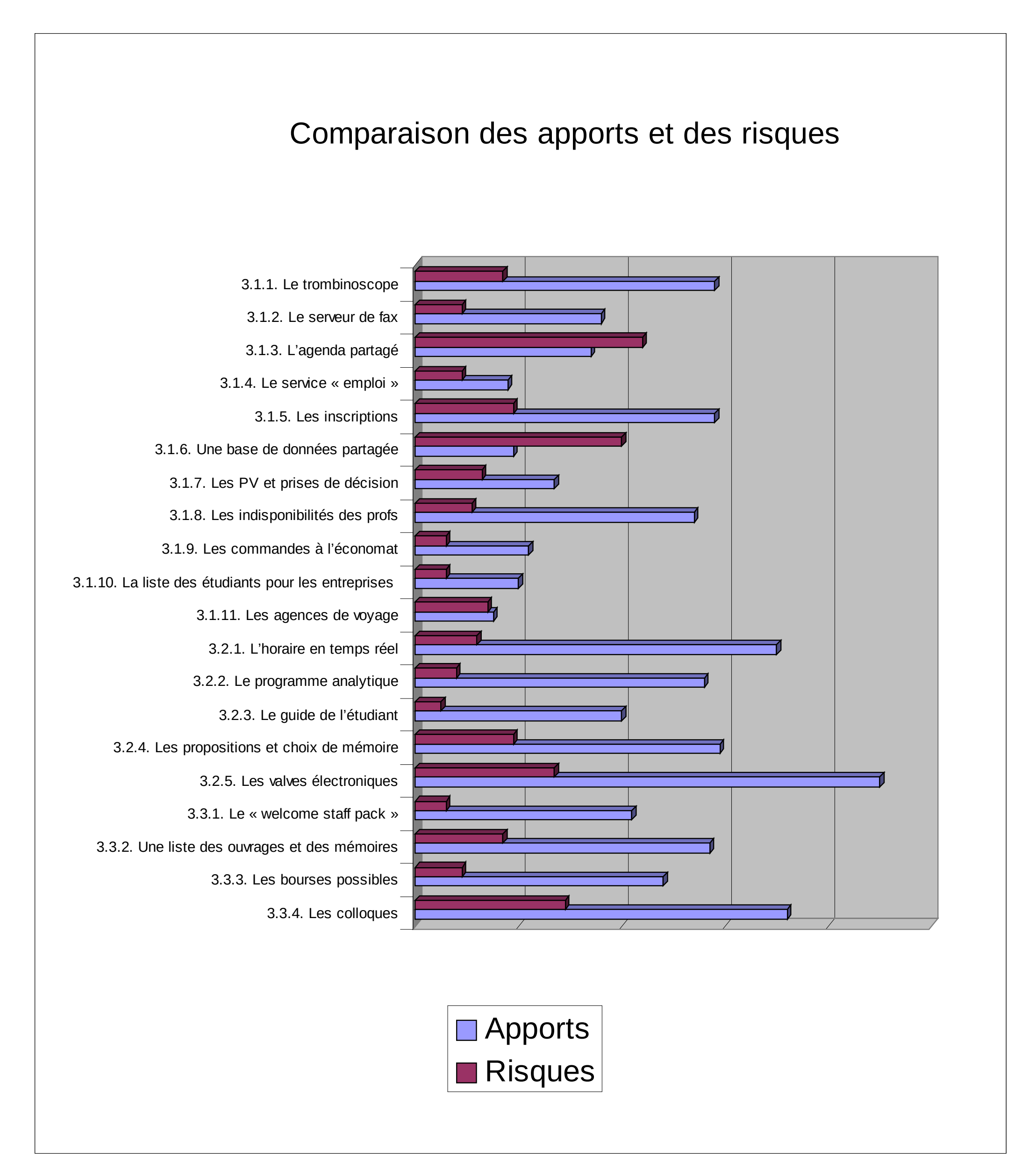


Figure 8 : Graphique de comparaison des apports et des risques
de l'ensemble des besoins

En tenant compte de l'intérêt que pourrait procurer chaque besoin et des risques techniques et organisationnels qu'il entraîne, nous avons élaboré une typologie, qui tente de situer chaque besoin en fonction du futur intranet de l'Institut.

II.4.3.1. Les projets abandonnés

Au vu de l'évaluation finale, deux projets semblent trop risqués par rapport au gain qu'ils apportent : l'agenda partagé, dont les risques organisationnels sont très importants, et la base de donnée partagée, trop peu utile compte tenu de l'effort technique à entreprendre. La relation avec les agences de voyage ne mérite pas plus d'intérêt. Puisque la publication des

procès-verbaux et des prises de décision des réunions diverses est déjà actuellement réalisée par le réseau et que ce service n'a pas reçu une évaluation importante, nous pouvons aussi l'abandonner. Quant au serveur de fax, on pourra facilement l'implémenter en dehors de l'intranet.

II.4.3.2. Les projets sous conditions

Le service « emploi » et la liste des étudiants pour les entreprises apportent finalement peu d'avantages, mais encourent aussi peu de risques ; ces services sont implémentables à condition de se combiner au sein d'une solution technique facile et rapide : un lien hypertexte sur le site Web de l'Institut, qui permettrait de contacter tous les étudiants de dernière année repris dans la liste (établie électroniquement) ; la sécurité serait minimalement assurée par une demande de confirmation par mail.

Les commandes à l'économat, vu leur peu d'intérêt global, mais aussi leur peu de risque, pourront être intégrées dans l'intranet à condition d'opter pour une solution technique rapide et facile.

II.4.3.3. Les projets à approfondir

Le trombinoscope apparaît comme une publication importante : nous ne pouvons toutefois la considérer comme facile. Pour mettre ce service en œuvre, une analyse complète de la technique devra être entreprise : faut-il une base de données ou un simple fichier HTML, comment accéder facilement et efficacement aux données d'un étudiant, comment intégrer les photographies au sein du trombino électronique…

La gestion des inscriptions pourra être intégrée dans l'intranet : la gestion relativement lourde de la mise à jour et de la disponibilité des formulaires doit cependant faire l'objet d'une analyse spécifique (qui acceptera d'être responsable, avec quels outils). Une étude plus poussée de la sécurité est également requise.

La gestion électronique des propositions et des choix de mémoire nécessite elle aussi un examen approfondi des implications techniques et organisationnelles : une base de données ou un traitement par fichier texte, une procédure entièrement automatisée ou un contrôle via le courrier électronique, une application globale ou des petits fichiers séparés…

La gestion des colloques, quant à elle, impose une très bonne connaissance de la démarche à suivre : un diagramme des flux sera, notamment, indispensable. Par ailleurs, il faudra absolument examiner les choix technologiques possibles, afin de choisir le plus efficace, le plus simple et le plus adapté.

II.4.3.4. Les publications faciles

Vu la facilité de réalisation et le peu de risque organisationnel encouru, l'intranet pourra rapidement permettre la publication électronique des documents suivants : le programme analytique des cours, la liste des ouvrages et des mémoires disponibles à l'Institut (à condition qu'elle soit établie), le « welcome staff pack » et le guide de l'étudiant.

II.4.3.5. Les priorités

Trois projets nous paraissent prioritaires : les valves électroniques, l'horaire en temps réel et la gestion des indisponibilités des professeurs pour les périodes de création d'horaires (de cours ou d'examens). La première proposition nécessite un traitement efficace et complexe de fichiers textes. Les deux autres font plutôt appel à une gestion distribuée de tableaux.

Conclusion du chapitre II

Ce chapitre a jeté les bases de la mise en place d'un intranet à l'Institut.

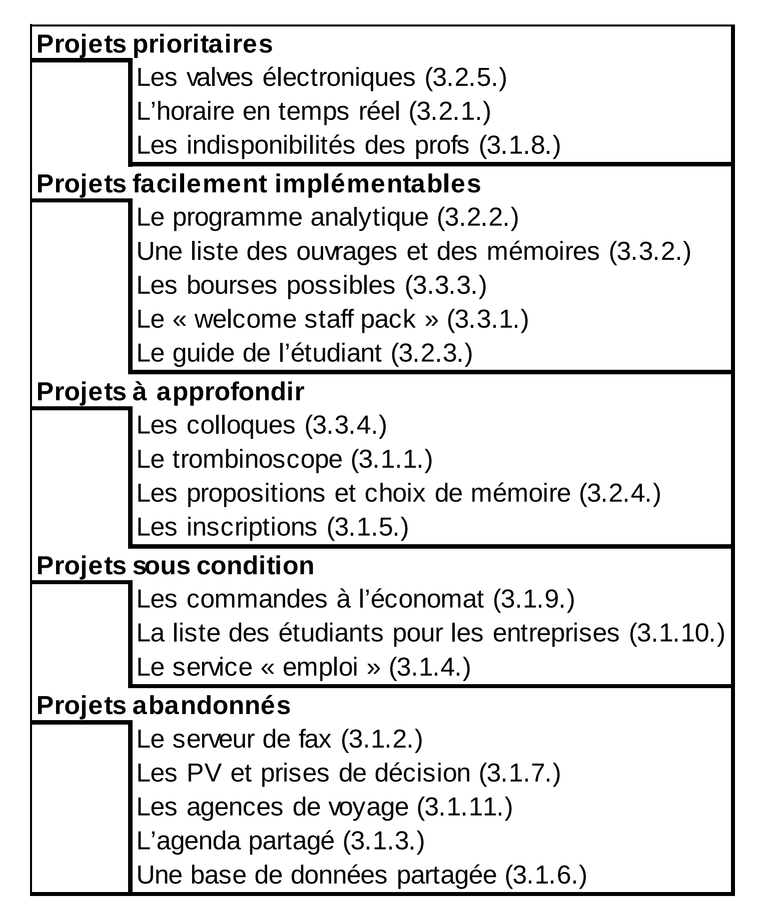
Tout d'abord, nous avons exposé quelques jalons théoriques inhérents à tout projet intranet, issus de l'expérience de consultants. Les pré-requis indispensables à l'avènement d'un Web interne tiennent dans quatre principes théoriques : une standardisation des composants du système d'information, un effort de centralisation des informations vers un serveur, une priorité absolue donnée à l'utilisateur dans les développements et les projets, et une orientation des procédures de traitement vers la communication. Une fois ces pré-requis mis en place, le projet intranet peut alors commencer à se déployer. Ce dernier doit commencer par une phase de préparation très soignée. Il s'agit, afin d'obtenir une vision claire du contexte de l'intranet, d'opérer une analyse des structures politiques, de l'infrastructure technique et des besoins fonctionnels. Grâce à cette préparation, les premières briques de l'Intranet pourront être particulièrement adaptées à l'organisation : une dynamique de développement incrémental peut alors être enclenchée. Trois phases clés jalonnent ce développement par incréments successifs : la mise en place des mécanismes de publication et de documentation, l'insertion des procédures de communication et de travail coopératif et enfin l'intégration d'applications distribuées interactives à proprement parler. Au sein de ces phases, une évaluation continue doit sans cesse être pratiquée.

Pour développer un intranet à l'Institut, nous devons donc confronter l'existant à ce cadre théorique. Grâce à une brève description de la situation technique et organisationnelle de l'Institut, nous pouvons constater que les pré-requis à l'intranet sont déjà en place : un bourgeon d'intranet est même déjà opérationnel et nous montre comment les aspects politiques, techniques et fonctionnels ont été traduits. Politiquement, l'Institut apparaît comme une bipartite, composée des membres du staff et des étudiants, où le pouvoir et les responsabilités restent concentrés dans des statuts relativement rigides. Au niveau technique, la simplicité et l'efficacité permettent d'assurer l'évolutivité des développements et l'intégrité d'un budget limité. Quant à l'aspect fonctionnel, l'intranet actuel nous montre un contenu hétéroclite et peu structuré, vraisemblablement dû au peu de matière fournie : une structure doit encore être construite.

De cette étude de l'existant, il ressort donc que la mise en place d'un intranet à l'Institut peut bénéficier de l'infrastructure et de l'ébauche de projet déjà mises en place, même si une structure générale doit encore être inventée. Dès lors, nous pouvons procéder à une étude des besoins communicationnels de l'Institut, en spécifiant les contraintes techniques et organisationnelles de chacun (l'aspect fonctionnel consistant dans la description du besoin) et en les structurant selon les pôles d'activité interne à l'Institut : l'administration, la pédagogie des cours dispensés et la recherche scientifique. Une fois la liste des besoins établie, un *scoring* nous permet de les évaluer et de les classer dans une typologie axée sur la

priorité de réalisation. Cette évaluation nécessite des critères, que nous nous forgeons autour des concepts « d'apport » et de « risque », et une méthode de cotation, pour laquelle nous choisissons de privilégier les apports d'un besoin sur les risques encourus.

Cette analyse des besoins débouche, finalement, sur les conclusions suivantes :



Projets prioritaires	
	Les valves électroniques (3.2.5.)
	L'horaire en temps réel (3.2.1.)
	Les indisponibilités des profs (3.1.8.)
Projets facilement implémentables	
	Le programme analytique (3.2.2.)
	Une liste des ouvrages et des mémoires (3.3.2.)
	Les bourses possibles (3.3.3.)
	Le « welcome staff pack » (3.3.1.)
	Le guide de l'étudiant (3.2.3.)
Projets à approfondir	
	Les colloques (3.3.4.)
	Le trombinoscope (3.1.1.)
	Les propositions et choix de mémoire (3.2.4.)
	Les inscriptions (3.1.5.)
Projets sous condition	
	Les commandes à l'économat (3.1.9.)
	La liste des étudiants pour les entreprises (3.1.10.)
	Le service « emploi » (3.1.4.)
Projets abandonnés	
	Le serveur de fax (3.1.2.)
	Les PV et prises de décision (3.1.7.)
	Les agences de voyage (3.1.11.)
	L'agenda partagé (3.1.3.)
	Une base de données partagée (3.1.6.)

Figure 9 : Typologie des besoins
selon leur priorité de réalisation pour l'intranet

Les projets facilement implémentables désignent en fait des simples publications de document au format HTML : ils encourent peu de risques et rendent des services plus ou moins appréciables. Leur concrétisation n'occupera ni temps ni effort. Les projets prioritaires puis les projets à approfondir constituent sans doute les composants de l'intranet de demain.

Dès lors, pour amorcer la dynamique incrémentale de l'intranet, nous allons, dans le chapitre suivant, procéder au développement de l'application considérée comme la plus intéressante : les valves électroniques.

Chapitre III. Les valves électroniques

Dans le premier chapitre, nous avons décrit en quoi consiste un intranet, l'infrastructure qu'il requiert, les méthodes de construction existantes, l'architecture type, les services qu'il rend et les conséquences organisationnelles qu'il peut impliquer. Le deuxième chapitre, pour sa part, a posé les bases d'un intranet à l'Institut : nous avons examiné si un tel projet était réalisable compte tenu du contexte et nous avons dressé une liste des besoins, avec leur priorité respective.

Dans ce troisième chapitre, nous tenterons de compléter concrètement le bourgeon d'intranet, d'ores et déjà présent à l'Institut, et ainsi d'enclencher une dynamique globale de développement, que d'autres pourront poursuivre dans l'avenir. Pour atteindre ce but, nous allons, dans un premier temps, nous pencher sur le besoin considéré comme le plus urgent par notre analyse : les valves électroniques. La liste des autres besoins forme, en quelque sorte, un premier cahier des charges du projet.

III.1. Les valves électroniques : quelques précisions

Pour construire cette application, nous allons procéder par étapes successives : tout d'abord, nous devrons préciser les besoins et les contraintes qui forment le projet. Ils ont déjà été brièvement exposés précédemment[76], mais il nous faut désormais les approfondir ; ensuite, nous concevrons une architecture conceptuelle ; puis, nous nous pencherons sur l'interface homme-machine de manière à élaborer des écrans conformes aux exigences ergonomiques que nous dégagerons ; enfin, nous aborderons les possibilités de maintenance et d'évolutivité de notre produit.

L'écriture du code ne sera pas traitée telle qu'elle dans ce chapitre : nous joignons, en annexe, la totalité de notre programme, afin que le lecteur intéressé puisse examiner nos choix concrets d'implémentation.

III.1.1. Précision des besoins

Pour réaliser les valves électroniques, nous ne pouvons nous satisfaire de l'étude des besoins effectuée dans le deuxième chapitre. Avant tout, nous devons la compléter pour assurer la meilleure base possible à notre développement.

---

[76] Cf. *supra*, section II.3.2.5., p. 55.

III.1.1.1. La situation actuelle

Les valves[77] désignent un tableau d'affichage, composé d'avis éclectiques. Ces avis, généralement, concernent des éléments ponctuels et possèdent donc une durée de vie limitée. Un avis est déposé pour un certain temps, il est ensuite retiré une fois que la période indiquée est achevée.

Les valves physiques (par opposition aux « valves électroniques ») sont structurées en fonction du groupe de personnes auquel elles s'adressent. Ainsi, les valves du staff ne sont pas situées au même endroit que celles des étudiants. Ces dernières sont elles-mêmes découpées en trois panneaux, conformément aux sections des 1ères Licences et Maîtrises, des 2èmes Licences et Maîtrises et des 3èmes Maîtrises. Chacun sait où il doit regarder, selon son statut ou selon la section à laquelle il appartient.

Vu leur position dans le bâtiment, les valves des étudiants peuvent être consultées, d'un simple coup d'œil, par tous les membres de l'Institut ; et tous les étudiants, quelle que soit leur année d'étude, peuvent lire les valves des autres années. Par contre, les valves du staff sont situées en un endroit qui leur est réservé, et ne sont donc pas visibles par les étudiants.

Actuellement, les valves destinées aux étudiants sont protégées par une vitre fermée à clef. Seule la secrétaire des étudiants dispose de cette clef ; elle seule a donc le droit d'y déposer un avis, d'en supprimer un… La gestion des valves du staff est moins stricte.

Soulignons également la présence de valves « officieuses », du moins pour les étudiants : les valves où sont affichées les offres d'emploi, les annonces d'activités estudiantines… Matériellement toutefois, celles-ci sont bien distinguées des valves officielles : elles ne sont pas protégées par une vitre fermée à clef (autrement dit, chacun peut venir déposer des avis et en retirer), elles ne sont pas situées au même endroit (les valves d'emploi sont à un autre étage, les valves d'annonce d'événements sont dispersées un peu partout) et, en général, elles sont moins bien tenues (des lambeaux de papier pendent continuellement à ces panneaux).

III.1.1.2. La migration électronique

A l'aube de l'intranet, la « migration » des valves vers des procédures électroniques représente à la fois un symbole et un pari. Un symbole d'une part, parce que le système des valves constitue un canal privilégié de communication au sein d'une institution académique comme la nôtre. D'autre part un pari, parce que ce service concerne de nombreux acteurs à l'Institut que le professionnalisme informatique rend exigeants et critiques à l'égard de toute application. Avec l'implantation de valves électroniques, il s'agit de convaincre les utilisateurs du futur intranet de son utilité, et de les impliquer dans un processus constant d'enrichissement de ce produit.

[77] Le terme « valves » est en fait un belgicisme. Il s'emploie toujours au pluriel et son genre est féminin. Toutefois, dans la suite de notre travail, nous l'emploierons de temps à autre au singulier, de manière à distinguer un panneau d'affichage des autres. Dans une logique de programmation, cette faute de langage permet, nous semble-t-il, plus de clarté.

Pour satisfaire les utilisateurs, le système électronique doit leur apporter des avantages que ne leur offre pas le système manuel. Et, inversement, les valves électroniques ne peuvent amoindrir la qualité habituelle du service des valves : ce seront les contraintes qui pèseront sur ce projet.

L'avantage décisif qu'apporte le service d'intranet sur le panneau d'affichage réside dans les possibilités d'automatisation des procédures de gestion des valves. L'ajout et la suppression d'un avis électronique s'opéreront par des traitements automatiques : il suffira d'introduire les éléments en question pour voir les choses « se faire toutes seules » ; plus besoin d'imprimer la page, de trouver la clef, de se déplacer et de coller le papier. Mais l'automatisation de la gestion des valves électroniques doit aussi concerner le « nettoyage » de celles-ci. Alors que les valves physiques exigent un entretien important, les valves électroniques, grâce à un système de date de péremption des avis, seront totalement autonomes. Il suffira d'introduire, lors du dépôt d'un avis, la date à laquelle celui-ci devra être supprimé. Ce nettoyage automatique des valves ne devra pas nous dispenser de prévoir une procédure de suppression d'avis, afin de résoudre facilement les cas d'erreurs ou de plaisanteries.

Pour encore accroître la facilité et la puissance du système électronique, nous proposons également de gérer automatiquement le dépôt d'avis aux valves ! Chaque avis disposerait, non seulement d'une date de péremption qui réglerait la question de l'entretien des valves électroniques, mais aussi d'une date de dépôt, qui aiderait à leur mise à jour. Il serait donc possible de déposer un avis sur les valves électroniques, qui devrait prendre cours à une date précise et être supprimé à une autre date précise. Bien sûr, il faudrait aussi pouvoir consulter la liste de ces avis « en attente », afin de les vérifier, et même pouvoir supprimer des avis en attente (parce que l'information a changé, par exemple). Cette date de dépôt constitue un des atouts majeurs des valves électroniques. Avec le double système de la date de dépôt et de la date de péremption, les valves électroniques paraîtront à la fois plus souples et plus puissantes que les valves physiques ; en tout cas pour les responsables actuels de la maintenance des valves, à savoir les secrétaires des étudiants pour les valves destinées à ces derniers.

En ce qui concerne les « lecteurs » de valves (à savoir tous ceux qui les consultent, sans nécessairement devoir ajouter ou supprimer des avis), le système électronique leur offre sa distribution de l'information : pour consulter les valves, il ne sera plus nécessaire de se déplacer à l'endroit du panneau d'affichage, mais un simple clic au bon endroit sur l'intranet suffira.

Ces avantages porteront leurs fruits à condition de ne pas être contrebalancés par des inconvénients. Il nous faut donc désormais examiner l'ensemble des inconvénients possibles afin d'en retirer les contraintes qui en découlent.

III.1.2. Les contraintes techniques et les choix de développement

III.1.2.1. L'application des principes généraux

Le chapitre précédent nous a exposé les principes clés du développement de l'intranet. Technologiquement, les applications doivent être standardisées (pour l'évolution fonctionnelle du projet), simples (pour l'évolution organisationnelle du projet) et efficaces (pour rester opérationnel et pour éviter des investissements techniques).

Dès lors, les axes directeurs de notre développement s'imposent : nous devons créer un CGI (*Common Gateway Interface*) spécifique, qui sera supporté par le serveur Unix de l'Institut. En effet, toutes les autres solutions nous paraissent inopportunes : les *applets* Java nécessitent trop de ressources, les contrôles ActiveX se limitent à l'environnement Windows 32 bits, le langage JavaScript n'est pas assez puissant. De plus, l'implantation d'une base de données nous semble contredire le principe de simplicité, puisque l'Institut ne dispose pas d'une base de données Oracle. Nous effectuerons, dès lors, une gestion de fichiers de type « texte ».

Pour un CGI en Unix, plusieurs langages sont possibles : C, qui présente l'avantage d'être quasi universel ; C++ ou TCL, qui offrent les potentialités d'une orientation objet ; Perl, qui peut à la fois tourner sur Unix et sur Windows NT ; AWK, qui est spécialement prévu pour effectuer des opérations sur des textes et des chaînes de caractère ; et enfin la programmation Shell (en Sh, Csh ou Bash), qui présente la syntaxe la plus facilement compréhensible. Toutes ces possibilités comportent du positif et du négatif ; dans l'ensemble, elles se valent. Nous avons choisi de développer les valves électroniques en programmation Shell, plus précisément en Bash, de manière à faciliter la lecture du code et donc la maintenance de l'application, qui sera réalisée par d'autres que nous. Afin d'alléger le plus possible la charge du serveur et du réseau, nous utiliserons aussi le langage JavaScript pour valider les données de formulaires. Grâce à ce langage orienté objet, nous pouvons déléguer au poste client la détection d'erreurs de cohérence de données et l'affichage de messages d'erreurs adéquats. Autrement dit, lorsque les données parviendront au serveur, celui-ci ne devra plus consacrer de ressources à la gestion d'erreurs éventuelles (principalement les transactions avec le poste client comme l'affichage de messages d'erreur, le nouvel affichage du formulaire…). Cependant, pour garantir une totale cohérence des traitements, nous pouvons également prévoir un contrôle des *inputs* au sein du script lui-même : certaines versions anciennes des navigateurs n'interprètent pas le code JavaScript et risquent d'envoyer au serveur des données incohérentes. De même, toute manipulation illicite qui tenterait de contourner le contrôle du formulaire serait ainsi déjouée. Les scripts Unix et JavaScript seront donc les outils de construction des valves électroniques.

III.1.2.2. La rapidité de consultation

Les valves physiques que nous connaissons aujourd'hui présentent une grande facilité de consultation. Cette caractéristique doit toujours être garantie par la voie électronique. Mais un double problème surgit : un ordinateur est indispensable et l'affichage doit être immédiat.

Le premier problème touche surtout les étudiants qui doivent utiliser un ordinateur du *pool*, tandis que les membres du staff disposent d'un ordinateur personnel dans leur bureau. Pour consulter les valves, l'étudiant doit tout d'abord trouver un ordinateur libre, ensuite

lancer un navigateur, charger le site Web de l'Institut et enfin se connecter sur l'intranet à la page adéquate. Cette procédure risque de prendre plus de temps qu'une simple lecture d'un panneau d'affichage. Pour diminuer cet intervalle de temps, il est indispensable réserver un ou deux ordinateurs à la consultation des valves électroniques et de les placer dans un endroit accessible (par exemple dans le couloir du troisième étage) ; sur ceux-ci, seul un navigateur serait disponible, et le *firewall* du Centre de Calcul limiterait les flux d'information au réseau interne des Facultés. L'Institut possède justement deux anciens modèles d'ordinateur (des processeurs de type 486) qui pourraient faire l'affaire. Cette solution règle la première partie du problème. Par ailleurs, dans l'avenir, il est envisageable d'installer un réseau de moniteurs au sein de l'Institut, sur lesquels les valves pourraient défiler en permanence.

Pour assurer un affichage rapide des valves au sein de l'intranet, nous devons, d'ores et déjà, opérer un choix concret de développement. Deux solutions sont possibles pour implémenter les fichiers concrétisant les valves électroniques. D'une part, chaque valve serait un fichier texte sans balises HTML, où les données seraient structurées dans des champs spécifiques. Pour consulter ces valves, il faudrait lancer un script, qui convertirait le fichier texte en fichier HTML et permettrait son affichage dans le navigateur ; la consultation des valves est alors une fonction à programmer. D'autre part, chaque valve serait un fichier HTML, complet, que le navigateur peut charger et afficher en interprétant les balises spécifiques : dans ce cas, la consultation des valves n'est pas une fonction à programmer.

La première solution assure que la valve affichée correspond bien à la valve « réelle », et non à une ancienne page conservée dans le tampon du navigateur. Par ailleurs, les cas problématiques d'une valve inexistante ou d'une valve vide peuvent être plus facilement traités par le script de consultation et donner une réponse adaptée à l'utilisateur. Mais, lorsqu'une valve contiendra de nombreux avis, le script mettra du temps à l'afficher. La deuxième solution tranche surtout par son efficacité : consulter une valve ne nécessite pas l'exécution d'un script. Cependant, cette solution ne traite pas aisément les cas d'une valve inexistante ou d'une valve vide : si le fichier HTML n'existe pas, le navigateur indiquera simplement une erreur due à l'URL, si le fichier HTML est vide, le navigateur affichera une page vierge ; dans les deux cas, l'utilisateur risque de rester perplexe. Puisque la rapidité d'affichage est très importante, la deuxième possibilité nous paraît plus indiquée pour implanter les valves électroniques. Cependant, puisque les valves risquent de subir de nombreuses modifications, il faut veiller à les afficher dans leur état réel, et non dans une ancienne version issue du tampon du navigateur : il nous faudra introduire du code JavaScript approprié, placé dans chaque page, pour assurer l'actualisation de la page à chaque affichage. Les cas limites de fichiers inexistants ou vides devront aussi être pris en compte. On peut faire en sorte que tous les fichiers correspondant aux valves soient toujours présents sur le serveur : une erreur d'URL ne pourra jamais survenir. On peut également régler le problème des fichiers vides : si une valve ne contient aucun avis, le fichier qui la concrétise contiendra seulement l'indication « valve vide ». Dès lors, l'utilisateur sera toujours fixé : soit une valve contient des avis, et il peut les consulter, soit une valve ne contient pas d'avis, et elle le lui indique clairement. Jamais il ne sera confronté à une URL fausse ou à une page vierge.

III.1.2.3. La sécurité des valves

Puisque les principes généraux du développement intranet seront respectés, et puisque la rapidité de consultation des valves sera garantie, nous pouvons nous pencher sur une dernière contrainte technique : la sécurité. Comment traduire, dans la version électronique des valves, la protection qui entoure les valves physiques actuelles des étudiants ? Ici, il n’est pas possible de fermer l’accès aux fichiers par une vitre et une clef !

Il nous semble que le système actuel, basé sur le tri des numéros IP des machines hôtes du réseau, peut convenir. Ce système peut protéger les fichiers des valves à la fois à l’extérieur et au sein de l’Institut. Ainsi, on peut filtrer l’accès aux valves à l’entrée même de l’intranet du staff ou de l’intranet étudiant. Les fichiers des valves destinées aux étudiants ne seront accessibles que depuis leur propre intranet, et donc inaccessibles de l’extérieur. Les fichiers des valves destinés aux membres du staff seront déposés sur l’intranet du staff, et seront donc inabordables via l’extérieur ou via l’intranet étudiant.

Toutes les contraintes techniques qui pèsent sur le projet des valves électroniques peuvent donc être assumées. Qu’en est-il des contraintes organisationnelles ?

III.1.3. Contraintes organisationnelles

III.1.3.1. La charge de travail

Les valves électroniques ne peuvent entraîner une surcharge de travail à l’Institut. Nous pensons particulièrement à la gestion de celles-ci. Si la gestion des valves électroniques devait s’avérer trop lourde, le système classique des valves « papier » ne serait jamais abandonné et le système électronique, jamais employé. L’automatisation de la gestion des valves électroniques relève donc d’une contrainte importante au niveau organisationnel.

Bien sûr, une période de transition, durant laquelle le système manuel et le système électronique cohabiteraient, est indispensable. Une telle innovation organisationnelle requiert un temps d’adaptation de chacun des membres de l’Institut. Cette période de transition entraînera fatalement plus de travail que le fonctionnement d’un seul de ces deux systèmes. Toutefois, le confort apporté par l’intranet peut convaincre rapidement les uns et les autres d’opter définitivement pour la solution électronique.

III.1.3.2. La distribution de l’information et des pouvoirs

La sécurité des valves assure une distribution de l’information cohérente avec le statut de chacun. Tout comme leur version physique, les valves électroniques autorisent, en effet, tous les membres de l’Institut, à consulter les valves des étudiants, mais restreignent la consultation des valves du staff à ce dernier. Puisque les valves électroniques permettront de déposer des avis « en attente », c'est-à-dire des avis qui seront déposés aux valves à une date spécifiée, il faut aussi régler la question de l’accès à cette liste. Il nous semble peu intéressant que les étudiants puissent consulter ces « valves virtuelles » composées d’avis en attente : l’information les intéressera en temps utile, c'est-à-dire au temps indiqué par l’auteur de l’avis. Ce sont plutôt les personnes habilitées à ajouter et à supprimer des avis (aussi bien des avis effectifs que des avis en attente) qui sont susceptibles de consulter les valves virtuelles :

par exemple, si quelqu'un dépose un avis qui doit être activé dans dix jours, il peut oublier l'avoir fait quelques jours plus tard, et vouloir vérifier que cet avis est bien en attente de publication. Dès lors, il nous faut désormais déterminer qui peut bénéficier du pouvoir d'ajout et de suppression d'avis.

Cette responsabilité de la gestion des valves prête à la discussion. Actuellement, seules les secrétaires peuvent déposer ou supprimer un avis aux valves officielles. Si l'on s'en tient aux caractéristiques politiques de l'Institut, telles que nous les avions dégagées au chapitre précédent[78], nous devons respecter cette découpe stricte des tâches et des responsabilités, ainsi que privilégier un contrôle centralisé dans les mains d'une personne. Ces caractéristiques nous étaient apparues comme les conséquences d'une organisation académique relativement rigide. Concrètement, cela signifie que l'ajout et la suppression d'avis sur les valves des étudiants resteraient l'apanage des secrétaires administratives.

Mais cette structure figée est-elle irrémédiablement installée ? L'intranet, avec tout d'abord les valves électroniques, ne serait-il pas l'occasion d'introduire une dynamique de distribution des tâches et des pouvoirs ? Qu'on nous comprenne bien ici ! Notre but n'est pas de donner plus de pouvoir aux étudiants, ou à d'autres personnes, ou d'instaurer un « *chaotic system* » à la manière de Steven Telleen. Mais, il nous semble que permettre aux membres du staff de déposer des avis sur les valves des étudiants et de les retirer contribuerait à alléger la tâche des secrétaires administratives, tout en améliorant le flux de l'information et l'efficacité de la communication. Un contrôle des valves électroniques serait toutefois confié à un *Webmaster*, qui pourrait intervenir techniquement en cas de problèmes graves.

Ainsi, les formulaires de dépôt d'un avis aux valves seraient disponibles via l'intranet du staff ; tout qui a accès à cet intranet aura donc aussi accès à ce formulaire. Ceux-ci concerneraient non seulement les valves actuelles, mais aussi celles composées d'avis en attente de publication (ce que nous avons appelé les « valves virtuelles »). Nous posons donc l'hypothèse que les membres du personnel useront de ces outils avec raison.

Quoi qu'il en soit, chaque avis serait « signé » par l'adresse IP de la machine qui l'a soumis, de manière à pouvoir exercer un contrôle *a posteriori*. Cette ouverture des valves à tous les membres du staff représente un risque. En effet, si quelqu'un parvient à s'insinuer dans l'intranet « staff », il aura toute liberté de publication sur l'intranet. Si ce risque devait s'avérer trop élevé, il serait toujours possible de faire marche arrière dans cette dynamique de distribution des tâches par rapport aux valves. Dans ce cas, il suffirait de réserver les formulaires de dépôt d'avis aux diverses secrétaires, en les installant sur leur PC respectif, et non plus sur l'intranet et, ainsi, de retrouver l'organisation actuelle.

III.1.4. Synthèse

L'ensemble de ces contraintes nous ont, d'ores et déjà, donné des impératifs très concrets quant à la réalisation de ces valves. Ainsi, nous savons que nous écrirons des scripts Unix, afin de gérer des fichiers HTML, qu'il y aura autant de valves que d'années d'études à l'Institut, avec en plus les valves du staff, et enfin, que les valves seront réelles (composées d'avis actuellement publiés sur l'intranet) et virtuelles (composées d'avis publiés sur l'intranet à une date spécifiée). Concrètement, nous pouvons donc compter huit fichiers à implémenter :

---

[78] Cf. *supra*, section II.2.2.2., p. 43.

Valves réelles	Valves virtuelles
Valves du staff	Valves du staff
Valves des 1ères Licences et Maîtrises	Valves des 1ères Licences et Maîtrises
Valves des 2èmes Licences et Maîtrises	Valves des 2èmes Licences et Maîtrises
Valves des 3èmes Maîtrises	Valves des 3èmes Maîtrises

Figure 10 : Tableau des valves à prévoir

Ces huit fichiers HTML devront se trouver en permanence sur le serveur. Et, s'ils sont vides, ils devront contenir une indication de type « Valve vide », ou « Ces valves ne contiennent actuellement aucun avis ».

La gestion des valves électroniques devra comporter plusieurs fonctions : la consultation des valves (réelles ou virtuelles), l'ajout d'un avis (sur des valves réelles ou virtuelles), la suppression d'un avis (au sein des valves réelles ou virtuelles), la mise à jour des valves réelles et virtuelles.

Il nous faut, désormais, approfondir ces indications en une analyse spécifique : nous allons tout d'abord établir le schéma entité-association de notre problématique, puis nous examinerons chacun des aspects de l'application à créer, afin d'en dresser une architecture qui structurera notre développement.

III.2. Le schéma entité-association

Afin de nous donner une vision claire et structurée de la problématique des valves électroniques, et de fixer une sorte de dictionnaire des données, nous proposons, sur la page suivante, un schéma entité-association. La présentation de ce dernier est conforme à celle prrésentée par le logiciel DB-Main qui est utilisé à l'Institut.

Pour plus de clarté, rappelons la signification des sigles employés ici. Une entité est représentée par un rectangle : son nom est indiqué en majuscules dans la partie supérieure du rectangle, ses attributs éventuels sont indiqués dans la partie centrale du rectangle en minuscules, les éventuelles contraintes qui s'appliquent sur l'entité sont stipulées dans la partie inférieure du rectangle. Les associations sont insérées dans des espèces d'hexagones. Sur les lignes qui relient les associations aux entités, on retrouve les rôles et leur connectivité propre. La relation de sous-typage est indiquée par un triangle : le sur-type est relié par une ligne épaisse, tandis qu'une ligne plus fine joint le(s) sous-type(s). La lettre contenue dans le triangle indique la propriété de sous-typage : P, lorsque les sous-types forment une partition du sur-type ; T, pour indiquer que l'ensemble des sous-types correspond au sur-type, sans nécessairement être disjoints. Enfin les contraintes exprimées dans la partie inférieure des rectangles sont caractérisées, ici, par la mention « exact-1 » (*exactly-one*) et par les deux rôles concernés, c'est-à-dire qu'un et un seul des deux rôles doit nécessairement exister : c'est la contrainte d'exclusion des rôles qui est ainsi formulée.

Une fois la syntaxe précisée, nous pouvons nous attacher au schéma lui-même. Tout d'abord, nous devons remarquer qu'il n'est ni exhaustif, ni canonique. Trois critiques peuvent lui être adressées : les entités ne possèdent pas d'identifiant, tous les attributs de toutes les entités ne sont pas indiqués et la partie concernant les valves officieuses de l'Institut n'est pas développée. La dernière remarque révèle en fait un choix de développement : nous avons décidé de ne pas traiter, dans les valves électroniques, les valves totalement officieuses ; par contre, les valves qui seraient à la fois officielles et officieuses, comme les valves du staff par exemple, sont traitées, puisque la relation de sous-typage entre les valves officielles et officieuses n'est pas disjointe (le triangle qui les relie à l'entité Valve_Institut contient un T et non un P). Quant aux deux autres critiques, nous avons décidé de ne pas indiquer leurs identifiants et de ne pas décrire toutes leurs entités pour préserver un maximum de lisibilité à notre schéma, qui ne traitera, dès lors, que des aspects utiles au développement de notre application.

Sur ce schéma entité-association, nous avons tenté d'exprimer les différences de pouvoir entre le staff et les étudiants. Ainsi, les membres du staff possèdent une valve officielle, c'est-à-dire une valve réelle et une valve virtuelle, sur lesquelles ils peuvent créer ou supprimer un avis. Par ailleurs, ils peuvent consulter l'ensemble des valves officielles, tant réelles que virtuelles. Les étudiants, quant à eux, sont répartis en cycle d'étude (1ère Licence et Maîtrise, 2ème Licence et Maîtrise, 3ème Maîtrise) ; chaque cycle d'étude possède une valve officielle, c'est-à-dire une valve réelle et une valve virtuelle. Les étudiants ne peuvent consulter que les valves réelles qui sont destinées à l'ensemble des cycles d'étude : ils ne peuvent ni consulter les valves officielles du staff, ni leurs valves virtuelles. Parmi les membres du staff, un *Webmaster* est désigné pour contrôler l'ensemble des valves officielles de l'Institut.

Page blanche

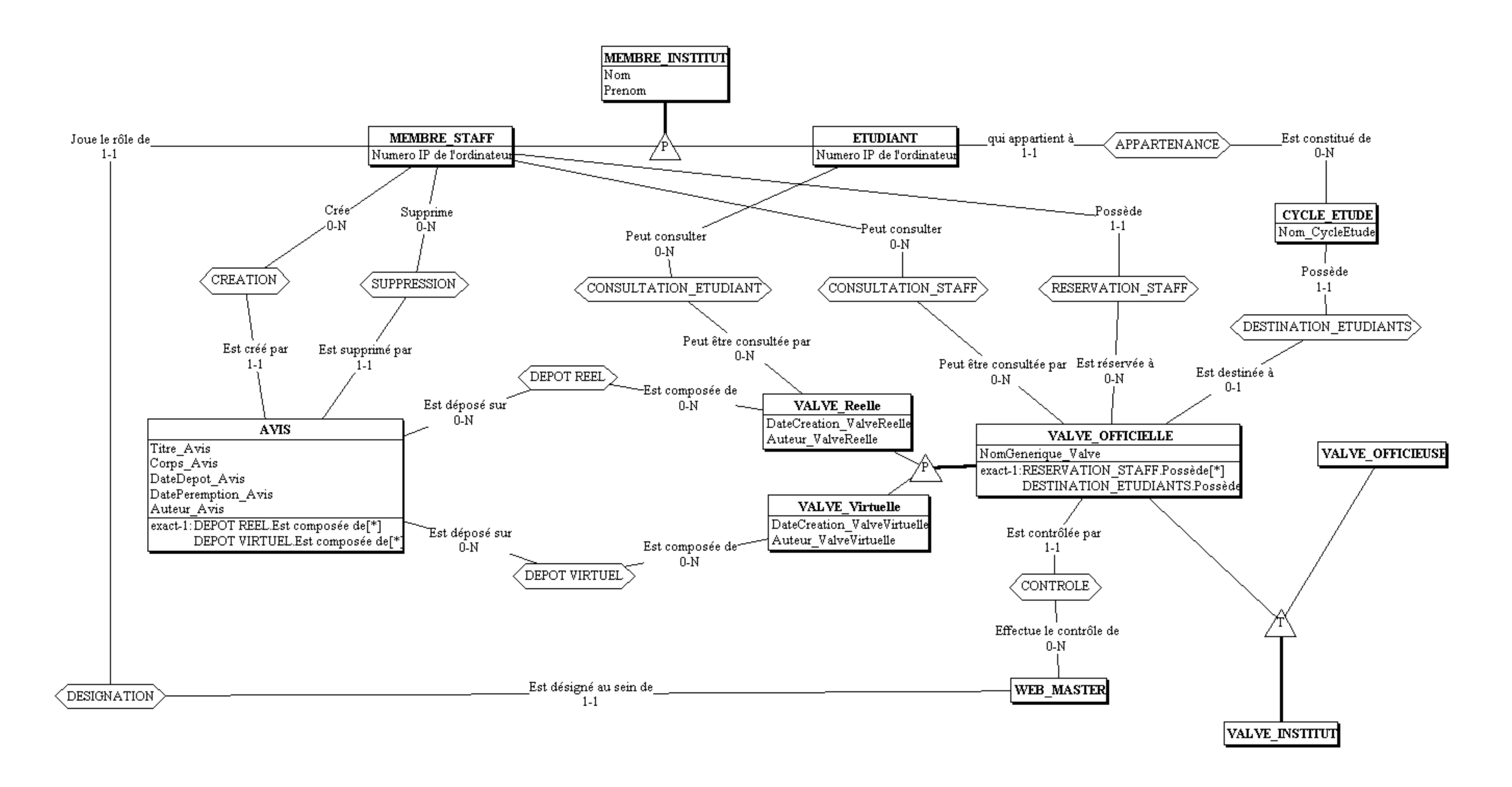


Figure 11 : Le schéma entité-association des valves de l'Institut.

III.3. L’architecture de l’application

Pour élaborer une architecture des fonctions de nos valves électroniques, nous allons utiliser une méthode personnelle. En effet, nous allons combiner deux types d’architecture pour dresser exhaustivement la liste des fonctions à créer et les structurer le plus clairement possible. Ainsi, nous allons tout d’abord concevoir une architecture orientée objet : celle-ci nous permettra de réfléchir en fonction des principales entités que le schéma ERA nous a fournies. Seules les entités ValvesOfficielles et Avis nous seront nécessaires pour notre développement : ce seront les objets (au sens de la programmation objet), contenus dans notre problème. Une fois que cette architecture objet nous aura éclairé sur les traitements à prévoir pour chaque entité, nous pourrons nous pencher sur leur agencement dans un programme impératif, composé de modules : une architecture plus conventionnelle, de type *top-down*, sera alors employée.

III.3.1. Spécifications des objets de l’application

La problématique des « valves électroniques » s’appuie sur deux entités fondamentales : l’entité « ValveOfficielle » et l’entité « Avis ». Puisque nous n’abordons que les valves officielles, nous nous contenterons de les appeler « valves ».

Nous avons donc besoin de deux objets : l’objet « valve » et l’objet « avis ». L’objet « valve » est plus général que l’objet « avis » : une valve est, en quelque sorte, l’ensemble ; et l’avis est l’élément. Chacun de ces objets comporte des propriétés, qui le caractérisent. Mais, ces objets, nous le savons d’ores et déjà, seront implantés concrètement sous la forme de fichiers HTML. Dès lors, nous devons aussi prévoir les marques HTML qui structureront ces fichiers. En combinant les attributs des entités, issus du schéma ERA, et les marques HTML nécessaires, nous pouvons détailler chacun des objets comme suit.

L’objet valve est structuré en fonction des propriétés suivantes :

- L’en-tête de la valve :
  - Le titre de la valve
  - La date de création de la valve
  - L’auteur de la valve
- Le corps de la valve, composé d’une suite d’avis (ou du message indiquant qu’elle est vide)
- La fin de la valve

L’objet avis contient les attributs suivants :

- L’en-tête de l’avis :
  - Le titre de l’avis
  - L’auteur (le numéro IP de la machine source)
- Le corps de l’avis
- La date de dépôt
- La date de péremption
- La fin de l’avis

Au sein des fichiers HTML, nous devons donc prévoir des indicateurs, qui nous permettront de distinguer les diverses propriétés de la valve et de l'avis. Ces indicateurs doivent être inclus dans le fichier comme des commentaires HTML (de manière à ne pas être affiché). Nous fixons ces indicateurs comme suit :

La valve		
	Marque de date de création de valve	<!-- Cree le ? ? ? -->
	Marque de créateur de valve	<!-- par le numero IP suivant ???-->
	Marque de fin d'en-tête	<!-- ======MarqueFinEntete====== -->
	Marque de fin de valve	<!-- ======MarqueFinValve====== -->
	Marque de valve vide	<!-- =====ValveVide===== -->
L'avis		
	Marque de début d'avis	<!-- =====MarqueDebutAvis===== -->
	Marque de début de titre	<!-- ====Titre==== -->
	Marque de début de contenu	<!-- ====Contenu==== -->
	Marque de début de date de dépôt	<!-- ====DateDepot==== -->
	Marque de début de date de péremption	<!-- ====DatePeremption==== -->
	Marque de début de nom d'auteur	<!-- ====Auteur==== de cet avis :??? -->
	Marque de fin d'avis	<!-- =====MarqueFinAvis===== -->

Figure 12 : Tableau des marques distinctives d'une valve et d'un avis

Quant aux balises HTML, nous veillerons à utiliser la présentation déjà utilisée aujourd'hui, pour les pages Web officielles de l'Institut. Ainsi, une valve pourrait avoir la forme suivante, avec ses indicateurs et ses balises HTML.

```
<HTML>
        <HEAD>
                <TITLE>Titre de la page</TITLE>
        </HEAD>
        <BODY BGCOLOR="#fffff0">
                <H1 ALIGN="CENTER">Titre des valves</H1>
                <hr ALIGN="LEFT">
                <!-- Cree le DateCreationValve -->
                <!-- par le numero IP suivant AuteurValve -->
        <!-- ======MarqueFinEntete====== -->

        <!-- =====MarqueDebutAvis===== -->
                <!-- ====Titre==== --> <H2>Titre de l'avis</H2></P>
                <!-- ====Contenu==== --> <BLOCKQUOTE>Contenu de l'avis
                        </BLOCKQUOTE></BR>
                <!-- ====DateDepot==== --> Jour / Mois / Année</P>
                <!-- ====DatePeremption==== --> Jour / Mois / Année
                <!-- ====Auteur==== de cet avis : AuteurAvis -->
                <hr ALIGN="LEFT">
        <!-- =====MarqueFinAvis===== -->

        <!-- ======MarqueFinValve====== -->
        </BODY>
</HTML>
```

Figure 13 : Exemple type d'un fichier de valve non vide

Dans cet exemple, nous n'avons inséré qu'un avis au sein de la valve, mais plusieurs avis peuvent cohabiter. Si la valve ne contenait aucun avis, elle porterait simplement, entre sa marque de fin d'en-tête et sa marque de fin de valve, la marque de valve vide. Pour être totalement clair dans la spécification des objets qui vont déterminer notre application, nous donnons aussi l'exemple d'une valve vide.

```
<HTML>
        <HEAD>
                <TITLE>Titre de la page</TITLE>
        </HEAD>
        <BODY BGCOLOR="#fffff0">
                <H1 ALIGN="CENTER">Titre des valves</H1>
                <hr ALIGN="LEFT">
                <!-- Cree le DateCreationValve -->
                <!-- par le numero IP suivant AuteurValve -->
        <!-- ======MarqueFinEntete====== -->

        <!-- =====ValveVide===== -->
                <H2> Aucun avis n'existe sur ces valves actuellement !</H2>

        <!-- ======MarqueFinValve====== -->
        </BODY>
</HTML>
```

Figure 14 : Exemple type d'un fichier de valve vide

Ces marques insérées dans les fichiers sont fondamentales. C'est grâce à elles que nous pourrons situer les diverses informations au sein d'un fichier de valves afin d'effectuer les traitements adéquats : par exemple, pour insérer un avis, il suffira de situer la marque de fin d'en-tête de la valve, et d'insérer l'avis juste après cette dernière ; pour supprimer un avis, il suffira de trouver ses marques de début et de fin ; pour vérifier les dates de dépôt et de péremption, il suffira de lire les données situées juste après les marques correspondantes.

Muni de la spécification des objets « valve » et « avis », nous pouvons nous pencher sur les diverses fonctionnalités qui leur correspondent.

III.3.2. Une architecture orientée objet

Les scripts Unix ne nous donnent pas la possibilité de programmer selon une méthode orientée objet. Seul un développement impératif est permis. Cependant, nous proposons d'établir une première architecture de notre application conformément à la méthode objet. Il nous semble, en effet, que cette méthode nous permettra de percevoir plus aisément les traitements à effectuer sur les valves et sur les avis, dans la foulée du schéma entité-association. Nous allons donc établir la liste des « méthodes » de nos objets. Une fois ces méthodes recensées, nous pourrons les structurer dans une architecture *top-down* plus conforme à la programmation impérative : ce sera l'architecture centrée sur les traitements.

III.3.2.1. Les méthodes de l'objet « avis »

L'objet avis sera, en fait, inséré au sein d'un objet plus générique, l'objet valve. Nous devons donc veiller à respecter la structure hiérarchique de ces deux objets, c'est-à-dire à ne pas demander à un avis des services qui relèvent d'un niveau hiérarchique supérieur, à savoir celui d'une valve. Par exemple, insérer un avis au sein d'une valve n'est pas du ressort de l'objet avis : parce qu'il lui est inférieur, un avis ne peut *utiliser* les méthodes de l'objet valve. Par contre, l'objet valve héritera des méthodes définies au sein d'un avis et pourra donc y faire appel.

Dès lors, un avis ne comptera que quelques méthodes essentielles. Puisque les données qui constituent un avis seront fournies par un utilisateur au moyen d'un formulaire HTML, il faut tout d'abord prévoir une méthode de décodage des données[79]. Ensuite, il faut une méthode de validation de celles-ci. Cette validation concerne particulièrement les dates : le jour, le mois et l'année des dates de dépôt et de péremption de l'avis doivent former un ensemble cohérent ; la date de dépôt doit être strictement antérieure à la date de péremption de l'avis, et la date de dépôt doit être postérieure ou égale à la date du jour. Remarquons d'emblée que la cohérence des dates doit concerner le nombre de jours dans un mois et le nombre de mois dans une année, avec la gestion particulière du mois de février et de l'année bissextile.

Une méthode qui discerne si l'avis est de type réel ou de type virtuel (c'est-à-dire si la date de dépôt est égale à celle de ce jour, ou si la date de dépôt lui est postérieure) s'avérera également très utile.

III.3.2.2. Les méthodes de l'objet « valve »

Les données de l'objet valve, au contraire de celles de l'objet avis, ne sont pas entrées par l'utilisateur. Il nous faut donc prévoir une méthode explicite de construction d'une valve (destinée à recevoir des avis). Puisque toutes les valves doivent toujours exister, quitte à être vides, une méthode spécifique de création de valve vide est nécessaire.

Dans la foulée, nous devons disposer de deux méthodes supplémentaires : nous avons à tester si une valve existe et si elle est vide (c'est-à-dire si elle ne contient aucun avis). Puisque les valves sont des objets génériques, nous devons également prévoir des méthodes d'introduction et de suppression d'avis. Afin d'assurer la mise à jour d'une valve, il faut encore une méthode qui détermine le traitement à lui appliquer (l'élimination d'avis périmés et/ou l'actualisation d'avis virtuels), une autre qui liste les avis périmés d'une valve, une autre qui répertorie les avis à déplacer de la valve virtuelle à la valve réelle (c'est-à-dire ceux dont la date dépôt est égale ou antérieure à celle d'aujourd'hui), une autre, enfin, qui copie les avis activables de la valve virtuelle à la valve réelle.

Pour pouvoir supprimer des avis, les valves doivent être affichées avec une mise en forme différente : le fichier HTML doit être modifié, afin d'être présenté comme un formulaire, dans lequel on peut choisir l'avis à supprimer. Cette présentation du fichier à supprimer via un formulaire pose cependant un problème important : pendant que l'utilisateur consulte le formulaire, le fichier peut être modifié par un autre utilisateur ; une méthode

[79] Pour décoder des données envoyées par HTTP à un CGI , on utilise généralement un cgi-parse.

spéciale devra vérifier la cohérence des valves consultées par rapport à la demande de suppression d'un avis. Puisque les valves sont des fichiers partagés, elles risquent d'être modifiées simultanément : une méthode qui ferme l'accès aux valves lorsqu'elles sont en modification s'impose donc. Cependant, nous devrons également veiller à ce qu'une erreur quelconque ne bloque pas continuellement les valves.

III.3.2.3. L'objet erreur

Afin de gérer les diverses erreurs qui peuvent survenir au cours d'un programme, nous allons aussi employer un objet « erreur ». Les méthodes de cet objet pourront être appelées par n'importe quel objet. Elles se chargeront essentiellement d'afficher les messages d'erreur adéquats, de donner à l'utilisateur l'un ou l'autre conseils et d'informer le *Webmaster* des problèmes graves par l'envoi d'un courrier électronique.

III.3.2.4. Représentation graphique de l'architecture objet

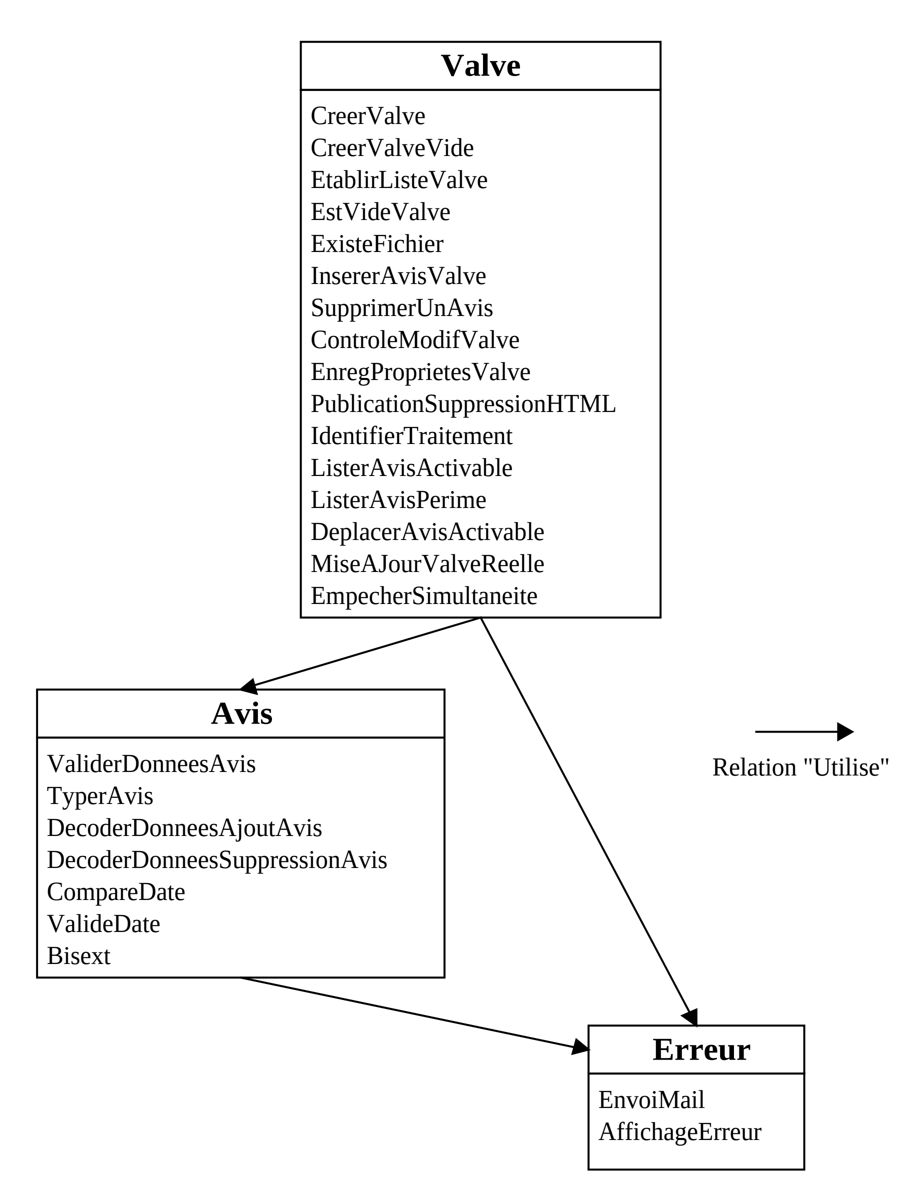


Figure 15 : Architecture orientée objet

III.3.3. Une architecture centrée sur les traitements

L'architecture orientée objet, telle que nous venons de l'élaborer, nous donne une première vision des traitements à effectuer sur chaque entité de notre application, particulièrement la valve et l'avis. Pour aller plus loin, nous allons abandonner la méthode objet et concevoir une architecture *top-down* orientée « traitement ». Cette utilisation de deux architectures n'est pas redondante : la première nous a renseigné sur les fonctions à créer autour des entités de notre problème, la seconde va nous aider à structurer ces fonctions dans des scripts, conformément aux exigences de la programmation Shell Unix.

Pour passer d'une architecture objet à une architecture classique, il nous suffit donc de reprendre l'ensemble des méthodes de nos objets (Avis, Valve, Erreur) et de considérer les relations de dépendance qui se tissent entre eux. Nous obtenons alors une hiérarchie de modules. Certains modules semblent plus généraux : ils ne sont appelés par aucun autre. Ce sont les modules qui forment le sommet de l'architecture *top-down* ; ils sont responsables des traitements les plus globaux et, pour les exécuter, font appel à d'autres modules. D'autres modules se situent plutôt au niveau intermédiaire : ils sont utilisés par d'autres et en utilisent eux-mêmes. Enfin, d'autres modules encore n'ont besoin d'aucun autre pour fonctionner : ce sont les modules terminaux, qui sont, en quelque sorte, des outils prêts à l'emploi. Ils forment la base de l'architecture *top-down*.

La problématique des valves électroniques requiert trois traitements globaux : l'ajout d'un avis, la suppression d'un avis, et la mise à jour des valves. Ce seront nos trois modules généraux. Nous pouvons d'ores et déjà penser que ces modules généraux prendront corps dans des scripts.

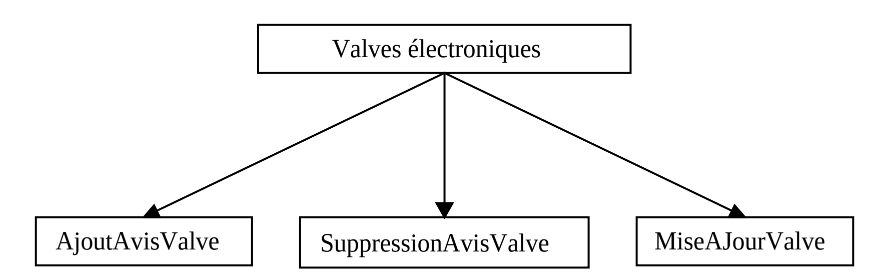


Figure 16 : Les modules généraux de l'architecture top-down

Concentrons-nous sur chacun de ces modules généraux afin de voir quels sont les modules intermédiaires et finaux qu'ils utilisent. Par la même occasion, tentons, d'ores et déjà, de percevoir la marche à suivre de chaque module général.

III.3.3.1. L'ajout d'un avis

Le point d'entrée de cette fonction résidera dans un formulaire HTML. Celui-ci comptera ses propres fonctions de validation des données, écrites en JavaScript. Lorsque ce formulaire sera validé, il déclenchera, grâce au *middelware*[80] HTTP, un script qui assumera la tâche d'ajouter un avis aux valves sélectionnées. Ce sera le script AjoutAvisValve, qui concrétisera le module correspondant. D'emblée, nous devons remarquer qu'un même avis peut être inséré auprès de plusieurs valves.

[80] Cf. *supra*, chapitre I, section 4.2. (p. 25).

Le script doit donc insérer l'avis introduit par l'utilisateur dans les valves spécifiées (au moins une). Tout d'abord, le script vérifiera que les valves ne sont pas actuellement en modification : lorsqu'un traitement est effectué sur une valve, le fichier « LockValve » est créé sur le serveur. Si ce fichier existe, il faut avertir l'utilisateur et lui demander de réessayer dans quelques instants, le script doit alors se terminer. Si ce fichier n'existe pas, le script doit le créer, afin d'indiquer à tout autre traitement que les valves sont en modification.

Pour renforcer la stabilité du programme, le script procédera, ensuite, à une validation des données envoyées au formulaire, pour le cas où le navigateur ne comprendrait pas JavaScript. Si les valves qui doivent recevoir l'avis sont inexistantes, le script les créera et enverra un *mail* au *Webmaster* pour l'avertir de cette situation anormale. Si les valves qui doivent recevoir l'avis sont vides, le script éliminera la marque de valve vide et la remplacera par l'avis à insérer. Enfin, si les valves comptent déjà des avis, le script insérera l'avis en question comme premier de la liste, c'est-à-dire juste après la marque de fin d'en-tête de chaque valve. Lorsque l'ajout sera effectué dans chaque valve demandée, le script libérera l'accès aux valves en supprimant le fichier témoin (LockValve).

Le script AjoutAvisValve peut donc être représenté par le graphique ci-dessous, par lequel nous avons tenté de traduire la séquence des diverses fonctions (une sorte d'algorithme général) et la structuration de ces fonctions entre elles (qui appelle qui).

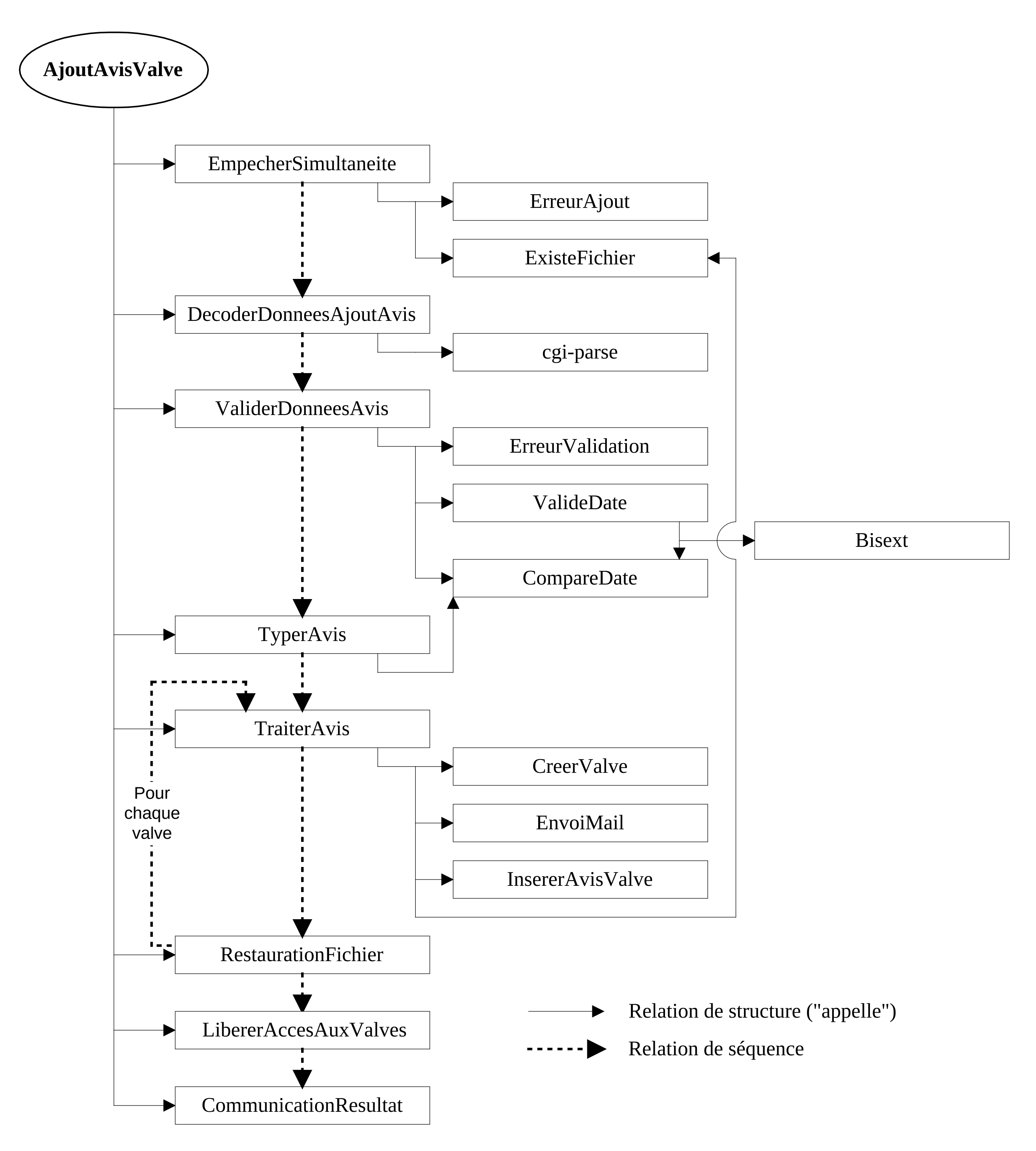


Figure 17 : Architecture du module général AjoutAvisValve

Le tableau suivant donne la liste des constantes et des variables globales mises en œuvre dans ce script.

	Nom	Valeur	Signification
Les constantes	CheminFichier	'../FichierValve/'	Répertoire où se situent les fichiers valves
	CstEntete	9	Longueur maximale de l'en-tête d'une valve
	CstLigne	12	Longueur minimale d'un avis
	Destinataire	www@info.fundp.ac.be'	Destinataire des mails (Webmaster)
	LockValve	CheminFichier + "LockValve"	Nom du fichier qui bloque les valves
	Sujet	ValveElectronique/ ControleAutomatique'	Sujet des mails
	URLInstitut	'http://www.info.fundp.ac.be'	URL citée en chaque fin de page Web
	URLIntranet	http://www.info.fundp.ac.be/ ~ffilee/intranet.html'	URL citée en chaque fin de page Web

	Nom	Type	Signification
Les variables globales	AnneeDepotAvis	Entier	Année de dépôt de l'avis une fois validée
	AnneePeremptionAvis	Entier	Année de péremption une fois validée
	AnneeSysteme	Entier	Valeur de l'année d'aujourd'hui (sur quatre chiffres)
	AuteurAvis	Chaîne de caractères	Auteur de l'avis
	ContenuAvis	Chaîne de caractères	Corps de l'avis une fois validé
	DestinataireAvis	Chaîne de caractères	Destinataire de l'avis
	FORM_AuAnnee	Chaîne de caractères	Année de péremption reçue du formulaire
	FORM_AuJour	Chaîne de caractères	Jour de péremption reçu du formulaire
	FORM_AuMois	Chaîne de caractères	Mois de péremption reçu du formulaire
	FORM_ContenuAvis	Chaîne de caractères	Corps de l'avis à ajouter reçu du formulaire
	FORM_DeAnnee	Chaîne de caractères	Année de dépôt reçue du formulaire
	FORM_DeJour	Chaîne de caractères	Jour de dépôt de l'avis reçu du formulaire
	FORM_DeMois	Chaîne de caractères	Mois de dépôt reçu du formulaire
	FORM_Ok1LM	Chaîne de caractères	Valeur du bouton radio Ok1LM du formulaire
	FORM_Ok2LM	Chaîne de caractères	Valeur du bouton radio OK2LM du formulaire
	FORM_Ok3M	Chaîne de caractères	Valeur du bouton radio OK3M du formulaire
	FORM_OkStaff	Chaîne de caractères	Valeur du bouton radio OkStaff du formulaire
	FORM_TitreAvis	Chaîne de caractères	Titre de l'avis à ajouter reçu du formulaire
	JourDepotAvis	Entier	Jour de dépôt de l'avis une fois validé
	JourPeremptionAvis	Entier	Jour de péremption une fois validé
	JourSysteme	Entier	Valeur du jour d'aujourd'hui
	MoisDepotAvis	Entier	Mois de dépôt de l'avis une fois validé
	MoisPeremptionAvis	Entier	Mois de péremption une fois validé
	MoisSysteme	Entier	Valeur du mois d'aujourd'hui
	NomFichInterm	Chaîne de caractères	Nom du fichier temporaire où seront recopiées les valves
	Ok1LM	Chaîne de caractères	Destinataire 1LM
	Ok2LM	Chaîne de caractères	Destinataire 2LM
	Ok3M	Chaîne de caractères	Destinataire 3M
	OkStaff	Chaîne de caractères	Destinataire Staff
	TitreAvis	Chaîne de caractères	Titre de l'avis une fois validé
	TypeAvis	Chaîne de caractères ('Reel' ou 'Virtuel')	Type de l'avis

Figure 18 : Tableau des constantes et des variables globales de AjoutAvisValve

III.3.3.2. La suppression d'un avis

Ce module ne peut se baser sur un formulaire préexistant. En effet, pour supprimer un avis d'une valve, il faut tout d'abord construire le formulaire, en fonction du contenu de cette valve. Une fois ce formulaire construit, il peut être soumis à l'utilisateur ; lorsqu'il est validé, il provoque l'exécution du script de suppression de l'avis sélectionné. Nous devons donc décomposer le module général SuppressionAvisValve en deux sous-modules : le module ConsultSupprValve et le module SuppressionAvis.

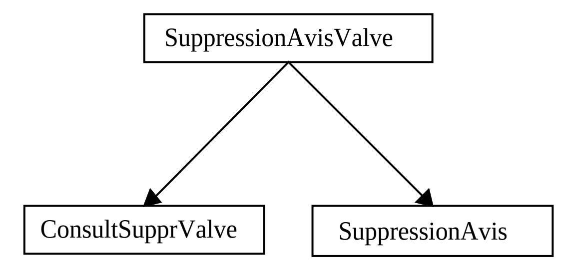


Figure 19 : Les deux sous-modules du module général SuppressionAvisValve

La suppression d'un avis s'effectue donc en deux temps : la construction du formulaire et la suppression de l'avis sélectionné dans le formulaire. L'utilisateur doit commencer par choisir la valve dans laquelle il veut supprimer un avis (la valve du staff, celle des 1ères Licences et Maîtrises, celle des 2èmes Licences et Maîtrises ou celle des 3èmes Maîtrises, qu'elle soit réelle ou virtuelle ; soit 8 possibilités), grâce à un lien hypertexte. Ce lien provoquera l'exécution d'un premier module, implanté dans un script, le script ConsultSupprValve. Ce script doit, tout d'abord, vérifier que cette valve existe. Si ce n'est pas le cas, il créera une valve vide, enverra un *mail* au *Webmaster*, et avertira l'utilisateur que la valve ne contient pas d'avis et donc, qu'il ne peut pas en effacer. Si la valve existe, le script vérifiera que l'accès aux valves est ouvert (c'est-à-dire que les valves ne sont pas en train d'être modifiées, par le fichier LockValve), puis il testera si la valve est vide ou si elle contient au moins un avis : dans le cas d'une valve vide, il se contentera d'avertir l'utilisateur de la situation ; dans le cas d'une valve non vide, le script enregistrera les caractéristiques de cette valve dans un fichier précis, ainsi que l'identité de l'utilisateur, puis il affichera la valve sous le format d'un formulaire HTML.

L'enregistrement des caractéristiques de la valve (résultat d'un « ls -l » en Unix) et de l'identité de l'utilisateur (en fait le numéro IP de la machine qu'il utilise) permettront de vérifier, lors de la suppression de l'avis, que le fichier de la valve n'a pas été modifié depuis sa consultation dans le formulaire de suppression d'un avis.

Le script ConsultSupprValve doit recevoir, pour exécuter la mise en forme de la valve, deux paramètres : le nom générique de la valve à afficher (ValveStaff, Valve1LM, Valve2LM, Valve3M) et le type de la valve (Reel ou Virtuel). Le déroulement du script ainsi que la structuration de ses fonctions peuvent être illustrés par le graphique ci-dessous.

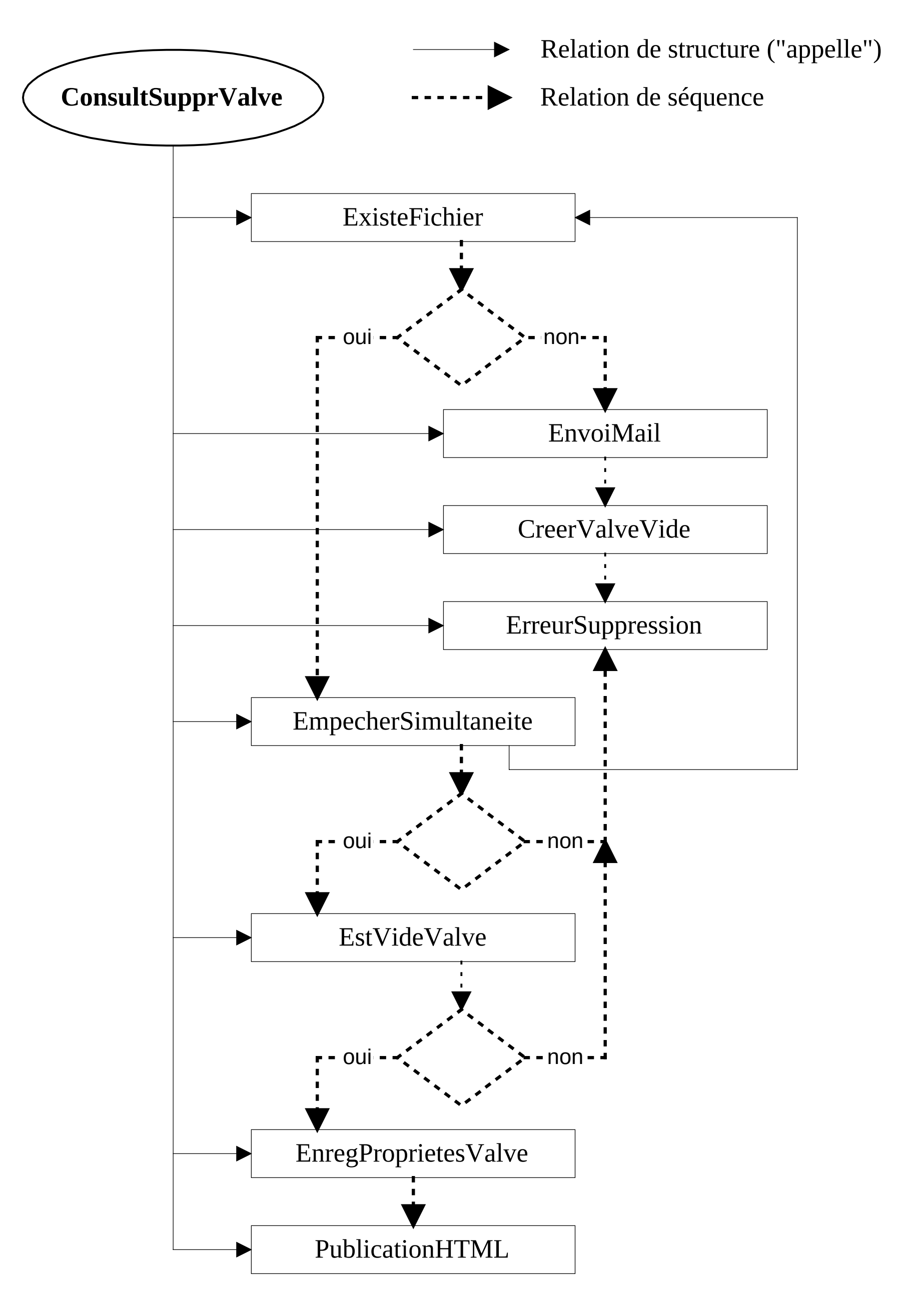


Figure 20 : Architecture du sous-module général ConsultSupprValve

Ce schéma peut être complété par une vision des constantes et des variables globales utilisées dans ce module.

	Nom	Valeur	Signification
Les constantes	CheminFichier	'../FichierValve/'	Répertoire où se situent les fichiers valves
	CstEntete	9	Longueur maximale de l'en-tête d'une valve
	CstLigne	12	Longueur minimale d'un avis
	Destinataire	www@info.fundp.ac.be'	Destinataire des mails (Webmaster)
	LockValve	CheminFichier + "LockValve"	Nom du fichier qui bloque les valves
	Sujet	ValveElectronique/ ControleAutomatique'	Sujet des mails
	URLInstitut	'http://www.info.fundp.ac.be'	URL citée en chaque fin de page Web
	URLIntranet	http://www.info.fundp.ac.be/ ~ffilee/intranet.html'	URL citée en chaque fin de page Web
	Nom	**Type**	**Signification**
Les variables globales	NomFichier	Chaîne de caractères (se termine par .html)	Nom du fichier des valves à afficher comme formulaire de suppression
	NomFichierPropriete	Chaîne de caractères (se termine par .Prop)	Nom du fichier qui reprendra les propriétés du fichier consulté, et le numéro IP de la machine du consulteur
	NomValve	Chaîne de caractères ('ValveStaff', 'Valve1LM', 'Valve2LM', 'Valve3M')	Nom générique d'une valve passé comme premier paramètre au script
	TypeValve	Chaîne de caractères ('Reel' ou 'Virtuel')	Type de valve passé comme deuxième paramètre au script

Figure 21 : Tableau des constantes et des variables globales de ConsultSupprValve

Le script ConsultSupprValve a donc créé un formulaire. Une fois celui-ci validé, c'est au tour d'un nouveau script de se déclencher : le script qui incarne le sous-module général SuppressionAvis. Ce dernier doit, tout d'abord, décoder les données transmises par le formulaire, via HTTP : il reçoit ainsi le nom de la valve, son type et le numéro de l'avis à supprimer. Les données décodées devront ensuite être validées, afin d'écarter toute fausse manœuvre ; au cas où les données n'apparaîtraient pas comme valides, un message d'avertissement serait envoyé à l'utilisateur. Puis, notre script va tester l'existence du fichier de cette valve. S'il n'existe pas (c'est-à-dire qu'il aurait disparu depuis l'exécution du script ConsultSupprValve), la procédure « d'urgence » est déclenchée : une valve vide est créée, un *mail* est envoyé au *Webmaster* et un écran d'avertissement est généré pour l'utilisateur. Dans le cas où le fichier existe encore, le script va bloquer l'accès aux valves (si possible), puis contrôler que ce fichier n'a pas été modifié depuis l'exécution du script ConsulSupprValve. Si ces tests ne sont pas concluants, le script se contentera de libérer l'accès aux valves et d'afficher un message adéquat à l'utilisateur. Dans le cas contraire, l'avis spécifié sera supprimé de la valve. Une fois la suppression réalisée, le script devra encore vérifier s'il reste au moins un avis dans la valve : dans ce cas, le traitement sera terminé ; sinon, le script remplacera la valve sans avis par une valve vide (c'est-à-dire qui contient une marque de valve vide). Graphiquement, ce script de SuppressionAvis donne la représentation suivante.

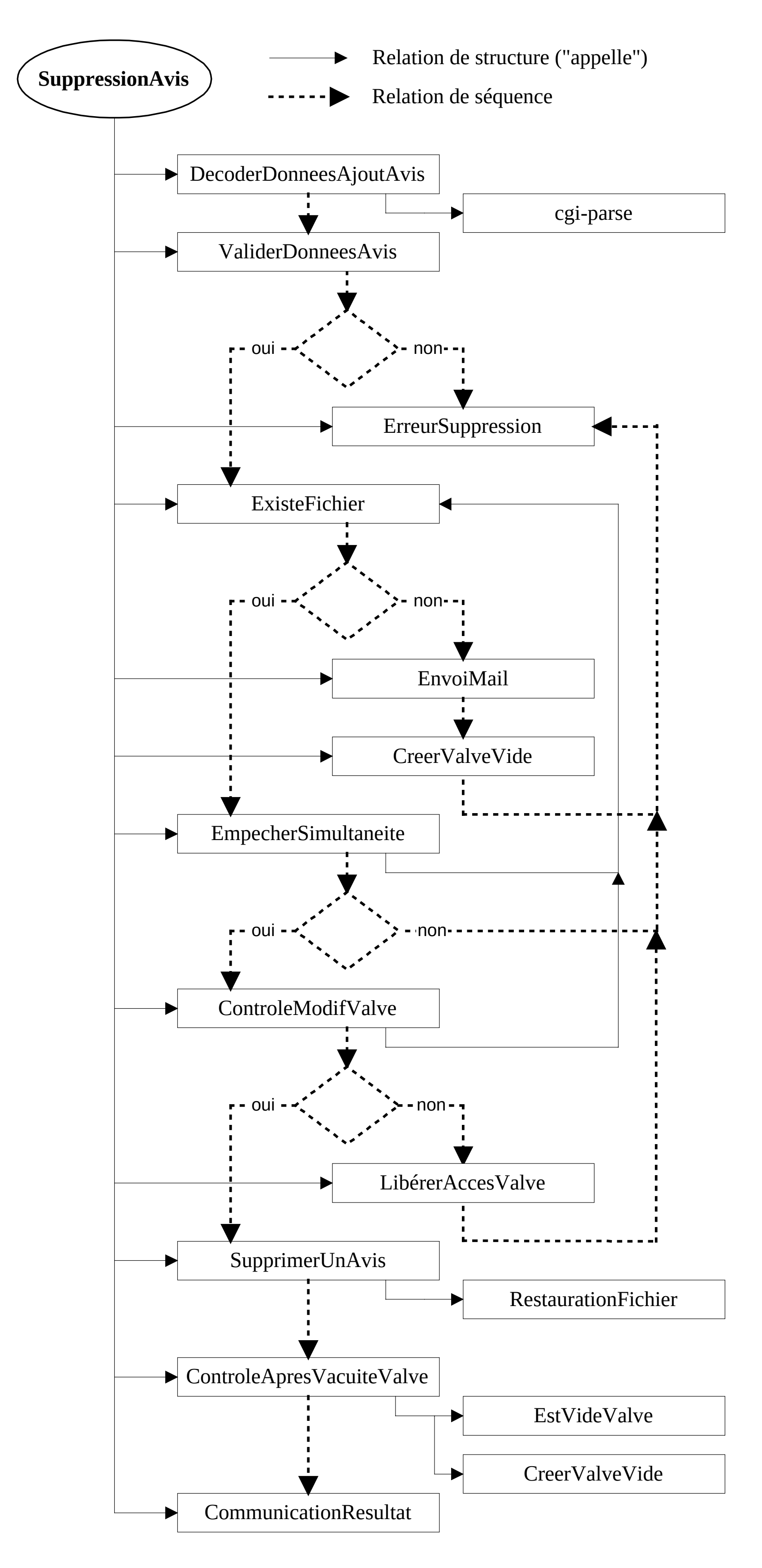


Figure 22 : Architecture du sous-module général SuppressionAvis

La liste des constantes et des variables globales de ce module peuvent être rassemblées dans le tableau suivant.

	Nom	Valeur	Signification
Les constantes	CheminFichier	'../FichierValve/'	Répertoire où se situent les fichiers valves
	CstEntete	9	Longueur maximale de l'en-tête d'une valve
	CstLigne	12	Longueur minimale d'un avis
	Destinataire	www@info.fundp.ac.be'	Destinataire des mails (Webmaster)
	ListeTypePossible	'Reel Virtuel'	La liste des types de valves possibles
	ListeValvePossible	ValveStaff Valve1LM Valve2LM Valve3M'	La liste des noms génériques de valves possibles
	LockValve	CheminFichier + "LockValve"	Nom du fichier qui bloque les valves
	Sujet	ValveElectronique/ ControleAutomatique'	Sujet des mails
	URLInstitut	'http://www.info.fundp.ac.be'	URL citée en chaque fin de page Web
	URLIntranet	http://www.info.fundp.ac.be/ ~ffilee/intranet.html'	URL citée en chaque fin de page Web
Les variables globales	**Nom**	**Type**	**Signification**
	FORM_NomValve	Chaîne de caractères	Nom générique d'une valve reçu du formulaire, sur laquelle la suppression d'un avis va s'exécuter
	FORM_NumAvisSupp	Entier	Numéro de l'avis à supprimer reçu du
	FORM_TypeValve	Chaîne de caractères	Type d'une valve reçu du formulaire, sur laquelle la suppression d'un avis va s'exécuter
	NomFichier	Chaîne de caractères (se termine par .html)	Nom du fichier des valves sur lequel la suppression va s'exécuter
	NomFichierPropriete	Chaîne de caractères (se termine par .Prop)	Nom du fichier qui reprend les propriétés du fichier consulté, et le numéro IP de la machine du consulteur
	NomValve	Chaîne de caractères ('ValveStaff', 'Valve1LM', 'Valve2LM', 'Valve3M')	Nom générique de la valve une fois validé, sur laquelle la suppression va s'exécuter
	NumAvisSuppr	Entier	Numéro de l'avis à supprimer une fois validé
	TypeValve	Chaîne de caractères ('Reel' ou 'Virtuel')	Type de la valve une fois validé, sur laquelle la suppression va s'exécuter

Figure 23 : Tableau des constantes et des variables globales de SuppressionAvis

III.3.3.3. La mise à jour des valves

Nous avons vu comment un ajout et une suppression d'avis peuvent être réalisés au sein de nos valves électroniques. Afin d'en terminer avec ce tour d'horizon, une dernière grande fonctionnalité doit encore être examinée : celle de la mise à jour des valves. Il s'agit, ici, de parcourir toutes les valves existantes afin de supprimer les avis périmés[81] au sein des valves réelles et de déplacer les avis activables[82] des valves virtuelles vers les valves réelles correspondantes. Le script MiseAJourValve assumera cette tâche.

Le point de départ de cette fonction ne tient pas dans un formulaire ou dans un lien hypertexte, comme c'était le cas pour l'ajout ou la suppression d'un avis. La mise à jour doit s'effectuer périodiquement (une fois par jour), de manière à garantir le nettoyage automatique

---

81 Les avis périmés sont ceux dont la date de péremption est postérieure à celle de ce jour.

82 Les avis activables sont ceux dont la date de dépôt est antérieure ou égale à celle de ce jour.

des valves et la parution des avis virtuels. Puisque nous travaillons sur un serveur Unix, nous pouvons confier l'exécution périodique de ce script au *Cron daemon*, c'est-à-dire à un processus qui tourne continuellement sur le serveur et qui active les programmes spécifiés au moment et à la fréquence souhaités. Nous pourrons demander l'exécution de la mise à jour des valves à une heure « creuse » sur le réseau, par exemple à 1h du matin.

La première tâche qui reviendra à ce script sera d'établir la liste des valves existantes ; cette dernière ne comptera que les noms des valves (ValveStaff, Valve1LM, Valve2LM, Valve3M) et non leur type. Lorsqu'une valve n'existe pas, le script la créera comme une valve vide et enverra un *mail* au *Webmaster*. Quand une valve n'existe, ni en tant que réelle, ni en tant que virtuelle, aucune mise à jour ne sera à prévoir ; le nom générique de cette valve ne sera pas inséré dans la liste des valves existantes. Par contre, si une valve existe en tant que réelle ou en tant que virtuelle, son nom générique sera repris dans cette liste. Dans le cas extrême où aucune valve n'existerait (aucun des huit fichiers), le script pourra s'arrêter là.

Lorsque la liste des valves existantes n'est pas vide, le script veillera à empêcher tout travail simultané sur les valves. Ce module doit être plus complet que dans les autres scripts. En effet, la présence du fichier qui indique une modification en cours des valves (le fichier LockValve) ne peut pas simplement entraîner l'arrêt du script de mise à jour : celle-ci doit se faire quotidiennement. Dès lors, le script vérifiera que le fichier en question a été modifié durant les trois dernières minutes : si ce n'est pas le cas, le script l'effacera, avertira le *Webmaster* par *mail* de cette présence inopportune, et continuera son exécution ; si le fichier a bel et bien été modifié durant les trois dernières minutes, le script se contentera de postposer son exécution de dix minutes (grâce à la fonction Unix « at ») et s'arrêtera.

Pour chacun des noms de valve repris dans la liste des valves existantes, il s'agira alors de procéder au traitement adéquat. Etant donné un nom générique de valve, le traitement sera identifié en testant si la valve réelle est vide ou si la valve virtuelle est vide : si les deux sont vides, aucun traitement n'est requis ; si seule la valve réelle est vide, le traitement consistera à simplement parcourir la valve virtuelle et à déplacer les avis activables vers la valve réelle (actualisation) ; si seule la valve virtuelle est vide, le traitement consistera à parcourir la valve réelle et à supprimer les avis périmés (péremption) ; si les deux valves ne sont pas vides, le traitement relèvera à la fois d'une actualisation et d'une péremption (traitement total).

En fonction du traitement identifié et étant donné le nom générique de la valve à traiter, le script relèvera le numéro des avis à supprimer dans la valve réelle et/ou relèvera le numéro des avis à déplacer de la valve virtuelle vers la valve réelle. La mise à jour correspondante (actualisation et/ou peremption) sera effectivement réalisée sur l'un et/ou l'autre fichier. Une fois la mise à jour effectuée, le script vérifiera, en conformité avec le traitement appliqué, si les fichiers impliqués contiennent, encore, au moins un avis : si un fichier devait s'avérer exempt d'avis, le script le recréerait comme une valve vide.

Le script pourra alors, dans la liste des valves existantes, passer au nom générique de valve suivant. Une fois la liste des valves existantes parcourues, le script pourra libérer l'accès aux valves.

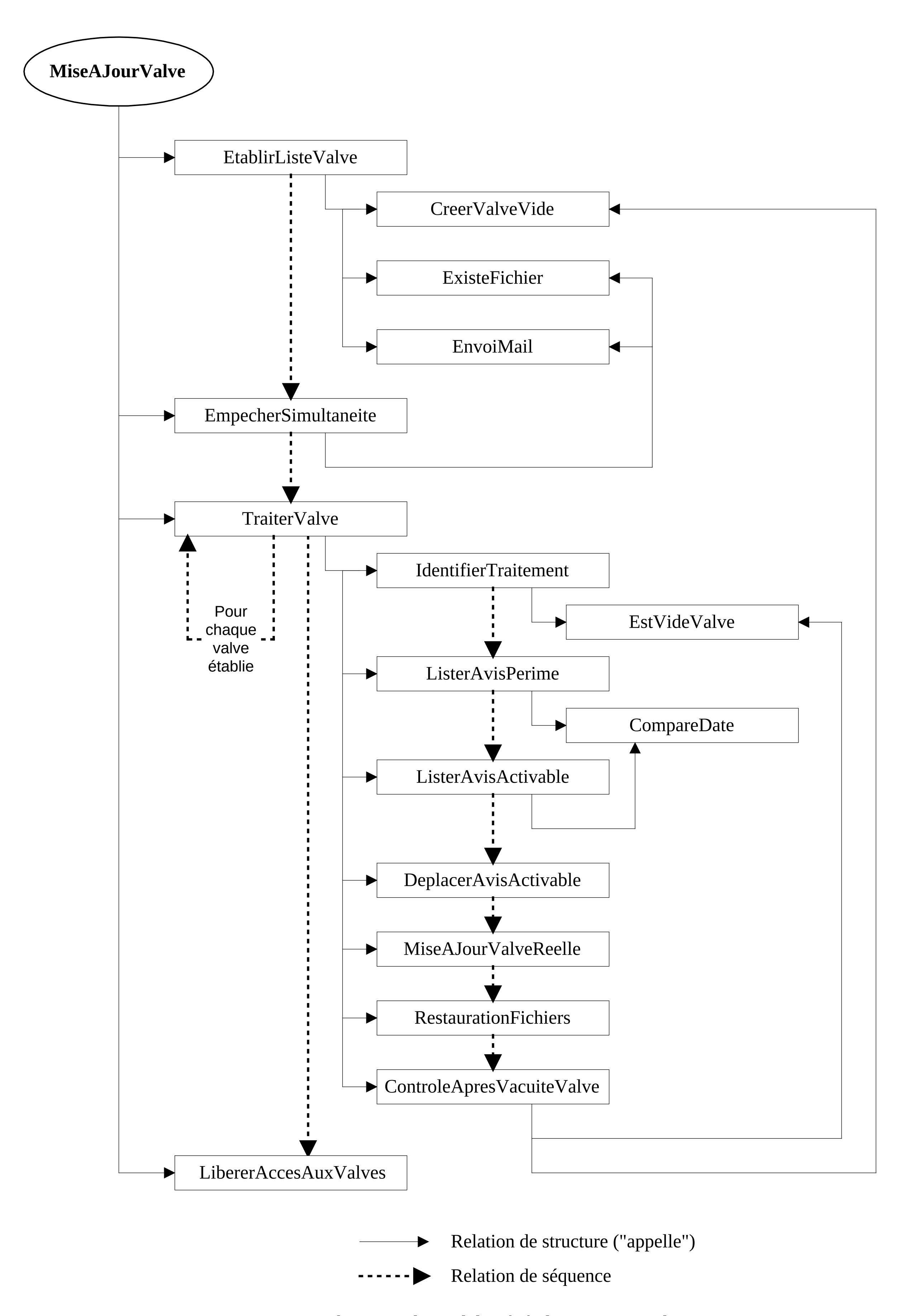


Figure 24 : Architecture du module général MiseAJourValve

Le module MiseAJourValve emploie les constantes et les variables globales que le tableau suivant nous détaillent.

	Nom	Valeur	Signification
Les constantes	CheminFichier	'../FichierValve/'	Répertoire où se situent les fichiers valves
	CstEntete	9	Longueur maximale de l'en-tête d'une valve
	CstLigne	12	Longueur minimale d'un avis
	Destinataire	www@info.fundp.ac.be'	Destinataire des mails (Webmaster)
	ListeValvePossible	ValveStaff Valve1LM Valve2LM Valve3M'	La liste des noms génériques de valves possibles
	LockValve	CheminFichier + "LockValve"	Nom du fichier qui bloque les valves
	Sujet	ValveElectronique/ ControleAutomatique'	Sujet des mails
	URLInstitut	'http://www.info.fundp.ac.be'	URL citée en chaque fin de page Web
	URLIntranet	http://www.info.fundp.ac.be/ ~ffilee/intranet.html'	URL citée en chaque fin de page Web
	Nom	**Type**	**Signification**
Les variables globales	ListeAvisActivable	Chaîne de caractères	La liste des numéros des avis "activables" (dont la date de dépôt est antérieure ou égale
	ListeAvisPerime	Chaîne de caractères	La liste des numéros des avis "périmés" (dont la date de péremption est strictement postérieure à celle d'aujourd'hui)
	ListeValveExistant	Chaîne de caractères	La liste des noms génériques de valves dont le fichier réel ou le fichier virtuel existe
	ListeValvePossible	ValveStaff Valve1LM Valve2LM Valve3M'	La liste des noms génériques de valves possibles
	NomScript	Chaîne de caractères	Le nom du script (en principe, MiseAJourValve.cgi)

Figure 25 : Tableau des constantes et des variables globales de MiseAJourValve

III.3.4. La répartition des modules dans les scripts

L'architecture orientée objet nous a montré les méthodes à prévoir pour chaque entité de notre problématique. Ensuite, une architecture davantage centrée sur les traitements nous a éclairé sur l'agencement de ces modules au sein de scripts linéaires.

Concrètement, nous débouchons sur la nécessité de construire quatre scripts, puisque la fonction de suppression d'un avis au sein d'une valve se décompose en deux :

- AjoutAvisValve
- ConsultSupprValve
- SuppressionAvis
- MiseAJourValve

Ces quatre scripts, comme nous l'ont illustré les schémas, vont employer des modules communs. Nous avons choisi de ne pas rassembler ceux-ci au sein d'une boîte à outils générale que nos quatre scripts pourraient appeler. Il nous semble, en effet, que cette solution, apparemment plus structurée, contribueraient à augmenter le nombre de scripts et finalement à obscurcir la lecture du code. Nous avons préféré construire nos quatre scripts indépendamment l'un de l'autre : chacun apparaîtra alors comme une unité bien distincte et bien lisible. Cette décision a un coût : certains modules seront répliqués dans plusieurs scripts.

Pour faciliter la maintenance, nous veillerons seulement à reproduire, mot pour mot, les modules communs dans chacun des scripts. Ainsi, pour modifier un module particulier, il suffit de le faire dans un script, et de répéter exactement la même correction dans les autres.

Une telle distribution des modules de l'application dans chacun des scripts nous impose de dresser une sorte de table des matières de nos valves électroniques. Grâce à cette table, on peut voir, rapidement, dans quels scripts se trouve tel module. Comme les modules seront classés par ordre alphabétique au sein de chaque script, au sein d'une rubrique intitulée « Déclaration des fonctions », les retrouver ne posera aucune difficulté. Le tableau suivant risque ainsi de s'avérer précieux pour l'évolution de l'application.

	Les scripts			
Les modules	**AjoutAvisValve**	**ConsultSupprValve**	**SuppressionAvis**	**MiseAJourValve**
Bisext	Oui			
CompareDate	Oui			Oui
ControleApresVacuiteValve			Oui	Oui
ControleModifValve			Oui	
CreerValve	Oui			Oui
CreerValveVide		Oui	Oui	Oui
DecoderDonneesAjoutAvis	Oui			
DecoderDonneesSupprAvis			Oui	
DeplacerAvisActivable				Oui
EmpecherSimultaneite	Oui		Oui	
EmpecherSimultaneiteCons		Oui		
EmpecherSimultaneiteMaj				Oui
EnregProprietesValve		Oui		
EnvoiMail	Oui	Oui	Oui	Oui
ErreurAjout	Oui			
ErreurSuppression		Oui	Oui	
ErreurValidation	Oui			
EstVideValve		Oui	Oui	Oui
EtablirListeValve				Oui
ExisteFichier	Oui	Oui	Oui	Oui
IdentifierTraitement				Oui
InsererAvisValve	Oui			
ListerAvisActivable				Oui
ListerAvisPerime				Oui
MiseAJourValveReelle				Oui
PublicationSupprHTML		Oui		
SupprimerUnAvis			Oui	
TraiterAvis	Oui			
TraiterValve				Oui
TyperAvis	Oui			
ValideDate	Oui			
ValiderDonneesAjoutAvis	Oui			
ValiderDonneesSupprAvis			Oui	

Figure 26 : Tableau de répartition des modules dans les scripts

L'architecture nous a montré quels modules programmer pour mettre en œuvre nos valves électroniques et comment les agencer dans des scripts pour garantir une lisibilité du code et une maintenance technique. Afin d'avancer d'un pas vers l'écriture du code, nous devons encore spécifier chacun de ces modules : la première annexe contient ainsi la liste de

tous les modules, présentés dans l'ordre alphabétique de leur intitulé. Pour chacun, sont précisés : les pré-conditions, les post-conditions, les modules qu'il utilise et une brève description du service qu'il rend.

Remarquons d'emblée que, dans un script Unix, une fonction ne peut que renvoyer un nombre entier comme code de retour (repris dans la variable $?). Pour qu'une fonction puisse renvoyer une information d'un autre type (chaîne de caractère, ensemble de nombres…), il faut employer une variable globale que la fonction modifiera. Lorsque des variables globales tiennent lieu de pré-conditions, nous l'avons indiqué précisément.

III.4. L'interface homme-machine

La conception de nos valves électroniques s'ébauche progressivement. En effet, le schéma entité-association nous a tout d'abord permis de structurer et de clarifier notre problème ; l'architecture nous a ensuite montré quels modules créer et comment les structurer. Pour la spécification de chaque module et l'écriture du code, nous renvoyons le lecteur dans les annexes.

Il nous semble encore important d'intégrer, dans l'élaboration conceptuelle de notre programme, les aspects qui concernent le contact avec l'utilisateur, c'est-à-dire l'interface homme-machine. L'IHM mérite une attention particulière, car elle constitue le lieu privilégié de la satisfaction ou de la frustration de l'utilisateur. Or, comme nous l'ont montré les premier et deuxième chapitres, l'utilisateur doit occuper la place centrale dans un projet intranet. Nous devons donc examiner les contraintes ergonomiques de notre application, afin d'en dégager les critères de conception de l'interface.

Nous avons déjà remarqué que nos valves électroniques devaient manifester une cohérence avec les pages Web qui forment le site de l'Institut et les quelques applications qui existent déjà sur l'intranet, comme la réservation du matériel ou des locaux. Cette cohérence se traduit bien concrètement par l'emploi d'une couleur de fond constante et par une mise en page relativement commune.

Pour aller plus loin, nous proposons de passer en revue les trois traitements qui appellent l'intervention de l'utilisateur : la consultation des valves, l'ajout d'un avis et la suppression d'un avis.

III.4.1. La consultation des valves

La consultation présente la particularité que les valves sont à elles-mêmes leur propre interface, puisque ce sont des fichiers contenant les balises HTML nécessaires à leur mise en page dans un navigateur. L'accès à ces valves sera simplement réalisé par des liens hypertextes à partir d'une page de l'intranet.

La contrainte ergonomique essentielle de la consultation de ces valves réside dans la rapidité d'exécution. Il faut que l'utilisateur puisse parcourir les valves rapidement, comme les valves physiques le permettent aujourd'hui. Pour garantir une rapidité d'affichage, nous avons construit notre architecture et nos spécifications de manière à accéder directement à des fichiers HTML. Afin de faciliter la lecture des valves, nous devons désormais prévoir une interface parfaitement claire : le titre de la valve doit apparaître sans équivoque, les avis

doivent être explicitement séparés les uns des autres, les avis les plus récents doivent apparaître en premier.

En basant nos valves sur des fichiers HTML, nous nous heurtons au problème du tampon du navigateur, dans lequel une page demandée demeure un certain temps. L'utilisateur qui consulte les valves devrait donc à chaque fois « rafraîchir » la page. Cette opération risque d'être omise, auquel cas l'utilisateur consulte une page qui n'est pas mise à jour. Et même, si elle est exécutée, cette opération prend un certain temps. En insérant un code JavaScript adéquat au début de chaque valve (code qui doit s'exécuter chaque fois que la page est affichée), nous permettons à l'utilisateur de trouver à la fois la rapidité de consultation et la sécurité d'obtenir une page à jour. De plus, pour faciliter la navigation au sein des diverses valves, nous pouvons aussi prévoir un lien hypertexte vers une page d'accueil générale en dessous de chaque valve. Compte tenu de ces contraintes, nous proposons l'interface suivante pour des valves :

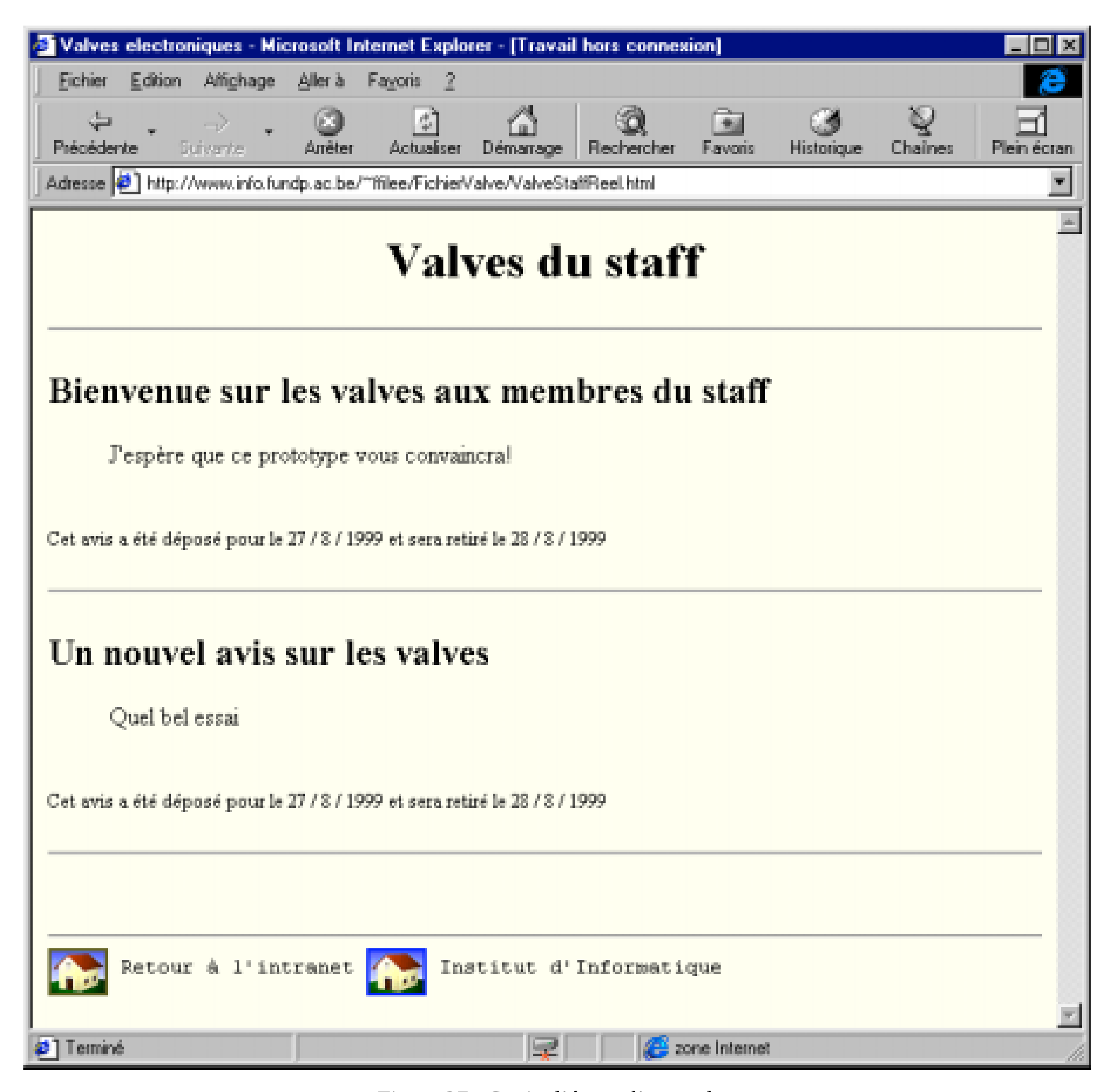


Figure 27 : Copie d'écran d'une valve

En haut de la page Web qui s'affiche, figure donc le titre de la valve. Si notre copie d'écran concernait la valve virtuelle du staff, nous aurions pu lire le titre suivant : « Valves du staff. Liste des avis en attente de publication ». Ensuite, se succèdent les divers avis, séparés par des lignes horizontales. Le titre de chaque avis apparaît en caractères gras, dans une police assez grande. Légèrement décalé et en plus petit, le corps de l'avis se situe juste en dessous du titre correspondant. Vient, enfin, la mention des dates de dépôt et de péremption. En fin de valve, on trouve le lien hypertexte qui permet de rejoindre une page d'accueil.

Nous avons aussi besoin de formulaires interactifs, dont la validation lance l'exécution du script correspondant sur le serveur. Deux modules généraux nécessitent un formulaire spécifique : l'ajout d'un avis à une ou plusieurs valves (effectif ou en attente) et la suppression d'un avis pour une valve déterminée (effective ou virtuelle).

III.4.2. Le formulaire d'ajout d'avis aux valves

Ce formulaire d'ajout aux valves est une page HTML qui réside constamment sur le serveur : un simple lien hypertexte suffit à l'activer.

Pour ajouter un avis aux valves (à une ou plusieurs valves), l'utilisateur doit entrer de nombreuses données : le titre, le corps, une date de dépôt, une date de péremption et un ou plusieurs destinataires.

Les contraintes ergonomiques ne sont plus dérivées de l'efficacité, comme dans le cas de la consultation : elles sont plutôt centrées sur la sécurité. Seul un avis cohérent peut être ajouté aux valves. Certes, nous avons vu que le script comportait des procédures de validation des données. Mais, puisque les ressources du réseau de l'Institut sont précieuses, nous devons les économiser. Dès lors, nous devons, autant que possible, traduire les contraintes au sein de l'interface elle-même : les échanges d'information entre le serveur et le poste client seront, dès lors, moins fréquents et la bande passante épargnée.

Dans les spécifications des modules, on trouvera le détail des contraintes d'intégrité des données d'un avis à ajouter aux valves[83]. Nous ne les citerons plus ici. Remarquons seulement que l'insertion des dates de dépôt et de péremption de l'avis peut-être éclairée par une découpe claire des zones de saisie en jour, mois et année. Pour servir d'exemple et, en même temps, retomber sur le cas le plus courant, la date du jour peut apparaître spontanément comme date de dépôt. Libre à l'utilisateur de la modifier !

Le contrôle de l'intégrité des données et la proposition automatique de la date du jour comme date de dépôt peuvent être réalisées grâce au langage orienté objet JavaScript. Un avis devient alors un objet, et les procédures de validation sont ses méthodes. Pour afficher la date du jour, les méthodes spécifiques à l'objet date peuvent être directement employées. Nous avons reproduit le code JavaScript de ce formulaire dans la troisième annexe.

[83] Cf. la spécification des modules ValiderDonneesAjoutAvis (cf. Annexes p. 14) et ValideDate (cf. Annexes p. 13).

Comme pour la consultation, on peut introduire, en bas de formulaire, un lien hypertexte vers la page d'accueil des valves. Voici l'interface que nous proposons pour le formulaire d'ajout d'un avis.

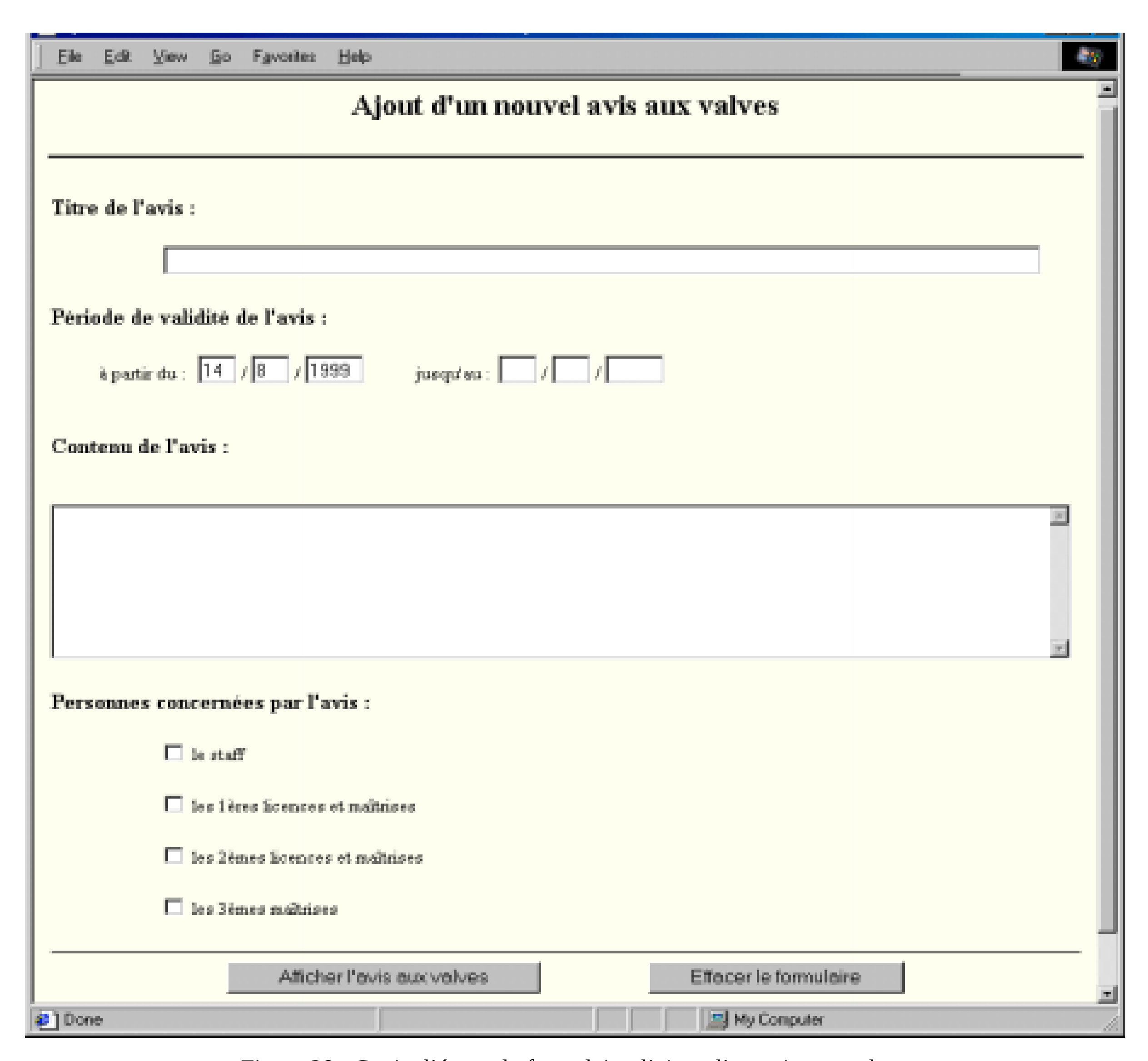


Figure 28 : Copie d'écran du formulaire d'ajout d'un avis aux valves

III.4.3. Le formulaire de suppression d'avis aux valves

Le dernier formulaire interactif de notre application concerne la suppression d'un avis. Ce formulaire n'existe pas comme fichier HTML. Il est construit par un script (ConsultSupprValve) en utilisant le fichier HTML de la valve et est envoyé au navigateur.

La mise en page de ce formulaire est donc, d'abord et avant tout, contrainte par le fichier de départ. Afin de ne pas effectuer de traitements trop importants, et pour ne pas retarder l'affichage du formulaire indéfiniment, le script va se contenter de rajouter quelques balises HTML sur certaines lignes du fichier des valves. Autrement dit, la mise en page du formulaire de suppression ne peut présenter une complexité ou une sophistication trop poussées. La simplicité de la mise en page ne constitue pas la seule exigence ergonomique.

Comme la fonction d'ajout d'un avis, celle de suppression doit garantir une sécurité absolue : le script SupprAvis requiert qu'un et un seul avis soit supprimé. Nous devons donc empêcher qu'aucun avis ne soit sélectionné dans le formulaire, ou que plusieurs avis y soient sélectionnés. Du code JavaScript n'est pas nécessaire pour vérifier ces contraintes : nous pouvons utiliser la propriété des boutons radios d'être mutuellement exclusifs lorsqu'ils portent le même nom. Si nous veillons à en initialiser un lors de l'affichage, la contrainte d'avoir toujours un avis sélectionné est totalement vérifiée. Pour transmettre au script SupprAvis le nom générique et le type de la valve concernée, le formulaire utilisera des champs cachés : non montrés à l'utilisateur, ces champs sont transmis comme tous les autres par HTTP au script. Dès lors, compte tenu de ces remarques, voici l'interface que nous proposons :

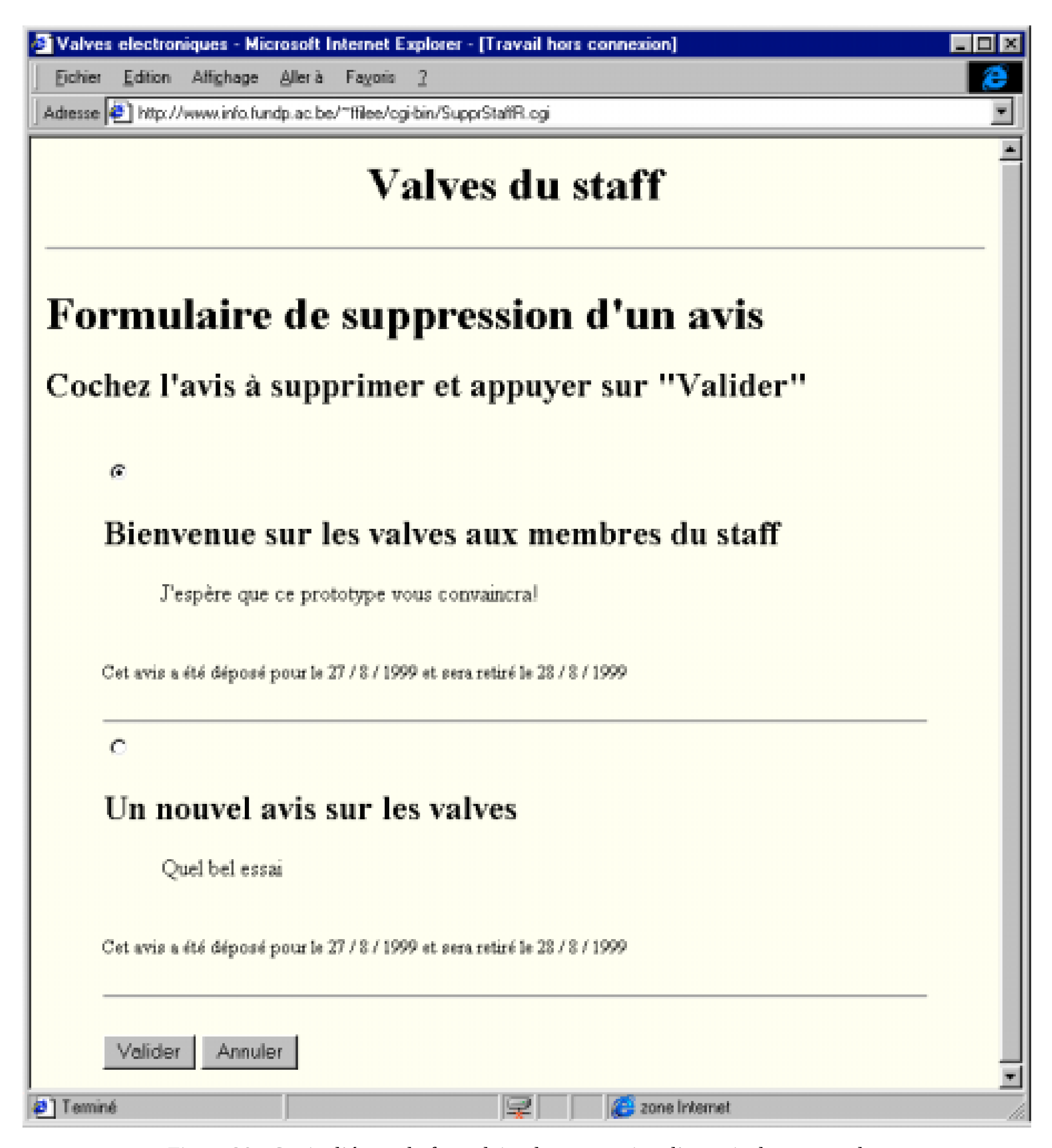


Figure 29 : Copie d'écran du formulaire de suppression d'un avis dans une valve

Avec ces trois écrans qui forment l'IHM des valves électroniques, les contraintes attenantes à chaque fonctionnalité nous semblent respectées : l'efficacité de consultation d'une valve, la sécurité d'ajout d'un avis, la simplicité de mise en page et la sécurité pour la suppression.

III.5. Les possibilités d'évolution

Nos valves possèdent désormais tout le nécessaire pour être réalisées : une architecture aux modules spécifiés, une interface homme-machine élaborée en cohérence avec les contraintes de chaque fonction interactive et une technologie d'implémentation d'ores et déjà choisie. Cependant, le développement des valves est loin d'être terminé. Bien sûr, la rédaction du code des scripts et les tests du programme doivent être effectués : les annexes reproduisent le code et on pourra se reporter à notre *home page* à l'Institut pour voir le programme tourner (http://www.info.fundp.ac.be/~ffilee/intranet.html). Mais, au-delà de ces opérations techniques, le développement des valves électroniques doit rester ouvert aux évolutions ultérieures.

En effet, puisqu'un intranet connaît un développement par incrément, nos valves doivent s'inscrire dans la même logique. Nous n'avons pas clôturé définitivement la migration des valves physiques vers les valves électroniques. Il s'agit plutôt ici d'un prototype avancé, qui amènera diverses remarques et sans doute de nouveaux besoins. Vraisemblablement donc, des modifications devront être apportées à notre logiciel.

Puisque ces modifications seront confiées à des personnes étrangères à la conception de ce programme, il nous a semblé utile de poser quelques indications pour faciliter cette maintenance évolutive. Ainsi, nous allons examiner cinq évolutions probables de nos valves.

III.5.1. Changer le *Webmaster* des valves

Sans doute le responsable des valves risque-t-il de changer de temps à autre. Or, les valves, en cas de problèmes relativement graves (détection de la présence inopportune du fichier LockValve ou absence d'un fichier de valves), envoient un courrier électronique à ce *Webmaster*. Changer le nom du destinataire de ces courriers s'avère une modification incontournable.

Celle-ci pourra être facilement réalisée. Le nom du destinataire des courriers électroniques, en effet, est enregistré dans une constante. Pour changer la valeur de cette constante, il suffit, dans chacun de nos quatre scripts, de retrouver la déclaration des constantes et des variables, qui se situe toujours après le code de chacune des fonctions, et de modifier l'affectation de la constante Destinataire. De la même manière, le libellé du sujet du courrier électronique peut être modifié via la constante Sujet.

Nous insistons pour que cette modification soit répercutée dans chacun des quatre scripts : la cohérence de notre application repose sur cette réplication du code des fonctions.

III.5.2. Diminuer le temps de réponse

Nous avons particulièrement soigné l'efficacité de notre application pour la consultation des valves, en créant directement des fichiers HTML. Les scripts d'ajout et de suppression d'avis, toutefois, peuvent manquer d'efficacité dans des cas extrêmes. En effet, lancés depuis le navigateur, ces scripts, dans le cas extrême de valves gigantesques, risquent de ne pas fournir à l'utilisateur une réponse suffisamment rapide : l'utilisateur aura tendance, dès lors, à interrompre le déroulement du script et à engendrer ainsi des situations incohérentes : seul le script de mise à jour des valves, lancé une fois par jour, est habilité à résoudre ces problèmes d'incohérence. Les valves risquent donc, au plus, une panne de 24 heures. Si le cas extrême de valves gigantesques devait se produire fréquemment, on pourrait alors mettre en œuvre un fonctionnement des scripts AjoutAvisValve et SuppressionAvis en deux temps.

Une fois les données validées, ces scripts pourraient, tout d'abord, envoyer la réponse d'acceptation de la requête à l'utilisateur, puis ils lanceraient l'ajout ou la suppression en tâche de fond sur le serveur, sans retour vers l'utilisateur. A l'inverse d'aujourd'hui, où le résultat n'est fourni à l'utilisateur qu'une fois le script achevé, ce dernier verrait assez rapidement si sa réponse est acceptée ou pas, mais ne pourrait constater immédiatement le résultat de sa requête sur les valves, puisque celles-ci seraient encore en traitement.

Ces modifications ne sont pas énormes à réaliser. Il suffirait de décomposer chacun de ces deux scripts en deux plus petits : le premier serait lancé par le formulaire HTML ; il se chargerait de la validation des données, lancerait le deuxième script en tâche de fond et communiquerait l'acceptation de la requête à l'utilisateur. Le deuxième script prendrait alors le relais en effectuant le traitement sur les valves à proprement parler. La communication entre les scripts pourrait s'effectuer via des paramètres (mais il faut savoir que ces derniers sont limités à dix en programmation Shell), ou via des fichiers intermédiaires.

Ce système a toutefois le désavantage de n'avertir l'utilisateur que sur la recevabilité ou non de sa requête. Il ne communique pas un résultat final. Par ailleurs, nous n'augmentons en rien l'efficacité des scripts ConsultSupprValve et MiseAJourValve. Pour le premier, le parcours séquentiel de la valve à afficher, même si elle est longue, s'impose absolument : pas question de donner un message non terminal à l'utilisateur ! Quant au second, il ne concerne pas l'utilisateur, puisqu'il s'exécute durant la nuit et n'envoie aucun résultat au navigateur.

Avant d'opérer ces modifications, il nous semble qu'il vaut mieux observer comment se comporte le système actuel en situation réelle.

III.5.3. Mieux gérer l'indisponibilité des valves

En situation réelle, les valves risquent souvent d'être modifiées. Peut-être faudra-t-il rapidement améliorer le dispositif de blocage et déblocage des valves. Nous proposons ici une double amélioration : d'une part une consultation de fichiers toujours cohérents, d'autre part la création d'une fonction consacrée à la gestion du fichier LockValve.

Lorsque les valves sont souvent modifiées, l'utilisateur risque d'accéder à un fichier des valves qui est justement en chantier : la consultation risque d'afficher un fichier incohérent ou incomplet. Pour remédier à cette carence, il suffit que les scripts qui modifient

les fichiers des valves (AjoutAvisValve, SuppressionAvis, MiseAJourValve) travaillent, non pas sur les fichiers originaux, mais sur des copies qu'ils effectueraient avant tout traitement. A la fin du script, il suffirait de remplacer les originaux par les copies. Lorsqu'il consulte une valve, l'utilisateur ne pourrait jamais tomber sur un fichier incohérent.

Le deuxième changement proposé concerne le fichier LockValve. Si de nombreux traitements sont effectués quotidiennement, la probabilité qu'un de ceux-ci n'arrive pas à terme est plus importante : lorsqu'ils ne vont pas jusqu'au bout, les scripts ne provoquent pas l'effacement de ce fichier qui bloque l'accès aux valves. Seul le script de mise à jour est muni d'un dispositif qui peut discerner cette présence inopportune et rétablir une situation viable : mais ce script n'est lancé qu'une fois par jour ! Les valves peuvent donc être « en panne » pendant une journée. Pour éviter cette situation, nous pourrions munir chacun des scripts d'un système similaire.

Puisque ces scripts travaillent en journée, au moment où les valves sont en effervescence, nous ne pouvons les mettre en attente pendant trois minutes, afin de voir si LockValve a été modifié durant cet intervalle, comme nous le faisons pour le script de mise à jour qui se déroule la nuit lorsque les valves sont au repos.

Mais, nous pourrions, lorsque LockValve a été détecté sur le serveur, arrêter le script et déclencher un script de contrôle de LockValve. Ce script, qui fonctionnerait en tâche de fond, pourrait vérifier si LockValve est modifié durant l'intervalle de temps défini (par exemple, trois minutes) et effacer ce fichier dans le cas contraire.

Cette modification de notre application n'a de sens que si les valves connaissent de nombreux ajouts et suppressions d'avis quotidiennement. Seule une observation du comportement des valves en situation réelle pourrait nous éclairer à ce propos.

III.5.4. Ajouter une nouvelle valve

Les valves électroniques, telles que nous les avons conçues, ne concernent que les membres du staff, les première et deuxième Licences ainsi que les première, deuxième et troisième Maîtrises. L'Institut, pourtant, compte d'autres sections : on pense notamment au D.E.A. et au D.G.T.I.C. Si notre prototype devait convaincre les membres de l'Institut, il faudrait vraisemblablement le compléter en lui ajoutant d'autres valves.

Cette opération n'est pas immédiate, mais n'est pas non plus d'une grande complexité. Elle requiert la modification de plusieurs fonctions au sein de notre application ainsi que celle du formulaire d'ajout d'un avis. Nous allons détailler ces modifications, en prenant comme exemple l'ajout de valves pour le D.E.A. :

- Dans l'IHM :
  - Le *formulaire d'ajout d'avis* doit comporter une nouvelle case à cocher avec les nouveaux destinataires possibles, dans la rubrique « personnes concernées par cet avis ». Comme les autres cases à cocher se nomment OkStaff, Ok1LM, Ok2LM et OK3LM, on pourrait intituler la nouvelle case à cocher OkDEA.

- Au sein des fonctions réparties dans les scripts :
  - La *fonction ValiderDonneesAjoutAvis* doit prendre en compte la nouvelle possibilité de valves lorsqu'elle affecte la variable DestinataireAvis.
  - La *fonction CreerValve* doit ajouter le cas OkDEA lorsqu'elle affecte la variable locale TitreValve, qui permettra d'introduire le titre adéquat dans le fichier valve.
  - La *fonction CreerValveVide* doit ajouter le cas OkDEA lorsqu'elle affecte la variable locale TitreValve, qui permettra d'introduire le titre adéquat dans le fichier valve.

- Dans la définition des variables et des constantes :
  - Le *script AjoutAvisValve* doit déclarer deux nouvelles variables globales : dans l'initialisation des variables du formulaire, il faut rajouter le nom générique de la valve tel qu'il sera décodé par la fonction de *parsing*, à savoir FORM_OkDEA initialisée à « OFF » (FORM_OkDEA='OFF') ; dans l'initialisation des variables issues du formulaire, il faut rajouter le nom générique de la valve initialisé à vide : OkDEA = ''.
  - Dans le *script SupprAvis*, il suffit d'ajouter à la variable ListeValvePossible le nom générique de la valve à ajouter : la ligne devient donc ListeValvePossible = 'ValveStaff Valve1LM Valve2LM Valve3M ValveDEA'
  - Dans le *script MiseAJourValve*, il suffit d'ajouter à la variable ListeValvePossible le nom générique de la valve à ajouter : la ligne devient donc ListeValvePossible = 'ValveStaff Valve1LM Valve2LM Valve3M ValveDEA'

Avec ces indications, l'ajout de nouvelles valves dans notre application ne devrait poser aucune difficulté. Vraisemblablement, il faut moins d'une heure pour réaliser ces modifications.

III.5.5. Différencier l'importance des avis

Pour sophistiquer notre application, nous pourrions également imaginer un système de différenciation d'avis selon leur importance. Par exemple, lorsqu'on insère un nouvel avis, on pourrait choisir s'il est très important, moyennement important ou d'ordre secondaire. Les avis seraient affichés dans les valves selon une double clé de tri : tout d'abord, on tiendrait compte de leur importance, de manière à afficher les plus essentiels au début des valves ; et ensuite, à l'intérieur de chacune des classes d'importance, les avis seraient triés selon leur date dépôt, de manière à afficher les plus récents d'abord.

Cette modification est relativement lourde à opérer et risque de ne pas s'avérer réellement utile : en effet, les personnes qui déposent leur avis sur les valves auront tendance à toujours considérer leur communication comme très importante ou moyennement importante, mais rarement comme secondaire. Toutefois, si le besoin se faisait malgré tout sentir dans l'avenir, nous tenons à indiquer les changements à opérer dans notre programme :

- Dans l'IHM :
  - Le *formulaire d'ajout d'avis* doit permettre d'entrer l'importance de l'avis à insérer dans les valves. Un ensemble de trois boutons radios mutuellement exclusifs (c'est-à-dire qui portent le même nom) et dont un est, par défaut, sélectionné (moyennement important) devrait, à l'aide de libellés significatifs, suffire. Ces boutons radios pourraient, par exemple, porter le nom « ImportanceAvis » et recevoir les valeurs « TresImportant », « MoyenImportant » ou « PasImportant ».

- Au sein des fonctions réparties dans les scripts :
  - La *fonction ValiderDonneesAjoutAvis* doit vérifier que la nouvelle variable globale FORM_ImportanceAvis corresponde bien à une des trois valeurs possibles. Une fois validée, elle en affectera le contenu à la nouvelle variable globale ImportanceAvis.
  - La fonction CreerValve La *fonction InsererAvis* doit subir de grosses modifications. Elle doit tout d'abord tenir compte d'une nouvelle marque à insérer dans la valve : <!-- ====Importance==== -->. Cette marque pourra être ajoutée après celles des dates de dépôt et de péremption, et juste avant celle de l'auteur. Afin d'être affichée dans le navigateur, la donnée d'importance devra être entourée de balises HTML. De plus, l'insertion d'un avis doit tenir compte de son importance et de l'importance de l'avis suivant, s'il existe. Il est nécessaire de réécrire le code de cette fonction.
  - De même, la fonction MiseAJourValveReelle doit être réécrite afin d'obéir aux mêmes exigences.

- Dans la définition des variables et des constantes :
  - Le *script AjoutAvisValve* doit déclarer deux nouvelles variables globales : dans l'initialisation des variables du formulaire, il faut rajouter le degré d'importance tel qu'il sera décodé par la fonction de *parsing*, à savoir FORM_ImportanceAvis initialisée à vide ; dans l'initialisation des variables issues du formulaire, il faut rajouter l'importance de l'avis intialisé à vide : ImportanceAvis = ''.

Puisqu'il est nécessaire de réécrire presque complètement deux fonctions très importantes, InsererAvisValve et MiseAJourValveReelle, la prise en compte d'un degré d'importance des avis doit être évaluée à l'aide d'une étude des besoins spécifique.

Cet examen de quelques évolutions possibles de nos valves électroniques pourrait bien sûr être poursuivi dans de nombreuses voies. Il nous semble, cependant, avoir envisagé les possibilités les plus réalistes. De plus, en donnant ces indications, nous avons montré comment procéder ; des modifications auxquelles nous n'aurions pas pensé pourraient s'inspirer de notre façon de faire.

Conclusion du chapitre III

Dans ce chapitre, nous avons commencé à réaliser l'intranet de l'Institut d'Informatique en développant l'application la plus intéressante, au vu du *scoring* établi dans notre deuxième chapitre : les valves électroniques.

Afin que ce point de départ constitue une base solide pour la suite du projet intranet, nous nous sommes efforcé de préciser le besoin et les contraintes qui le déterminent. Les valves nous sont ainsi apparues comme un ensemble d'avis temporaires, structurées selon le groupe de personnes auxquelles elles s'adressent, facilement consultables et protégées lorsqu'elles sont officielles : elles forment un canal de communication sûr et efficace pour l'ensemble de l'Institut.

Pour que la migration des valves vers des solutions électroniques rencontre le succès escompté, elle doit s'imposer comme plus avantageuse : les dates de dépôt et de péremption associées à chaque avis, le nettoyage automatique des valves et la distribution de l'information sur le réseau forment ses atouts. Cependant, les valves électroniques ne peuvent se contenter de ces possibilités nouvelles : avant tout, elles doivent offrir un service aux qualités équivalentes de celui qui est proposé actuellement par les panneaux d'affichage. Nous avons donc cerné ces qualités afin d'en déduire des contraintes techniques et organisationnelles.

Au niveau technique, nous avons souligné trois contraintes : la nécessité d'employer des outils standards, efficaces et simples, comme un CGI écrit avec des scripts Shell (par exemple en Bash) ; l'importance de conserver la rapidité de consultation des panneaux physiques, en installant des ordinateurs consacrés à la consultation des valves et en accédant directement à des fichiers HTML ; et l'exigence d'assurer une protection des valves électroniques, en appliquant le système de sécurité actuel de l'intranet basé sur le tri des numéros IP.

Quant aux contraintes organisationnelles qui pèsent sur la transformation électronique des valves, elles sont au nombre de deux : l'impossibilité d'augmenter la charge de travail des membres de l'Institut, ce que la gestion automatique des valves électroniques nous permet d'éviter autant que possible ; et le refus de laisser les valves aux mains de tous, tout en acceptant que les membres du staff puissent disposer des fonctions d'ajout et de suppression d'avis aux valves. L'ensemble de ces contraintes nous livre ainsi quelques principes de mise en œuvre : huit fichiers HTML, correspondant aux valves réelles et virtuelles du staff, des premières Licences et Maîtrises, des deuxièmes Licences et Maîtrises et des troisièmes maîtrises, formeront la matière sur laquelle les fonctions de consultation des valves, d'ajout d'avis, de suppression d'avis et de mise à jour de valves viendront effectuer leurs traitements.

Une fois les contours généraux de notre application dessinés, nous avons entrepris le développement à proprement parler. La première étape d'un développement de logiciel consiste à élaborer le schéma entité-association du problème à cerner : diverses entités, avec leurs attributs, ont donc vu le jour, reliées entre elles par des associations. A partir de ce schéma, nous avons réalisé une architecture qui renseigne à la fois sur les modules à créer et sur la structuration de ces modules entre eux. Pour atteindre ce double objectif, nous avons procédé en deux temps. Tout d'abord, une architecture orientée objet nous a éclairé sur les divers traitements à envisager pour les entités essentielles que le schéma entité-association

nous a données : l'objet valve, l'objet avis et l'objet erreur ont ainsi acquis leurs méthodes propres, afin de leur appliquer des opérations spécifiques. Dans un deuxième temps, nous avons repris les divers modules identifiés par l'architecture objet pour les insérer dans une architecture plus conforme aux principes de la programmation Shell en Unix. Cette architecture *top-down* a ainsi structuré ces modules selon la nature des traitements à effectuer, permettant, dès lors, une programmation impérative sous la forme de scripts. Nous avons ainsi découpé notre programme en une liste de modules prestataires de service, dont l'implémentation concrète ne peut plus réellement poser de difficulté.

La conception plus précise des valves électroniques a été détaillée dans la première annexe. Chaque module y a été spécifié avec ses pré-conditions, ses post-conditions, les modules qu'il utilise et une description plus libre de son fonctionnement. Le code de chacun des scripts, pour sa part, a été reproduit dans la deuxième annexe.

La dernière étape du développement des valves sur l'intranet relève du lien fondamental à établir avec l'utilisateur : c'est l'interface homme-machine qui forme l'intermédiaire entre les scripts que nous venons de décrire, et l'utilisateur dont le rôle central a été clairement posé dans les deux chapitres précédents. Trois traitements nécessitent une interactivité avec l'utilisateur : la consultation des valves, l'ajout d'un avis et la suppression d'un avis. Pour chacune d'entre elles, nous avons déterminé les contraintes ergonomiques et élaboré une interface qui nous semble les respecter.

Ainsi, la consultation requiert une efficacité importante, qui doit être concrétisée par une rapidité d'affichage et une clarté de disposition des avis. Les fonctions d'ajout et de suppression d'avis exigent toutes deux une sécurité importante : l'interface doit veiller à envoyer les données les plus cohérentes possibles aux scripts, de manière à restreindre les échanges entre le serveur et le client et ainsi à économiser les ressources du réseau. De plus, le formulaire de suppression d'avis doit être mis en page très sommairement, afin de diminuer le temps de traitement du script. Avec la réalisation de l'IHM, nos valves électroniques ont acquis leur dernière pièce maîtresse. Tout est rassemblé, désormais, pour que cette application tourne et rende les services qu'on attend d'elle.

Mais, croire que ce développement est achevé une fois pour toutes relève d'une utopie. Un intranet est toujours créé selon une logique incrémentale : il en va de même pour chacun de ses composants. Nos valves électroniques doivent donc prêter le flanc à des modifications ultérieures, qui constituent la dynamique évolutive : bien plutôt qu'un résultat achevé, nous proposons un prototype. Afin de faciliter l'évolutivité de ce dernier, nous avons, dès lors, envisagé quelques modifications et donné des indications concrètes de réalisation. Ainsi, nous avons établi la marche à suivre pour changer le *Webmaster* des valves, pour diminuer le temps de réponse des scripts d'ajout et de suppression d'avis, pour mieux gérer l'indisponibilité des valves en accédant à des fichiers toujours cohérents et en évitant les blocages des fichiers, pour ajouter de nouvelles valves à celles qui existent déjà et pour prendre en compte le degré d'importance des avis. Ces indications ne sont évidemment, ni exhaustives, ni nécessaires. Chacune des modifications envisagées doit être évaluée en fonction du comportement du prototype en situation réelle et des nouveaux besoins ressentis face à celui-ci. Bien loin d'être achevées, nos valves nécessitent donc une maintenance continue. L'intranet ne cessera jamais de se développer.

Page blanche

Conclusion générale

La théorie des organisations a depuis longtemps reconnu l'importance de l'information, et l'informatique s'est rapidement imposé comme le meilleur moyen de gérer rationnellement cette dernière : l'automatisation d'activités, via l'élaboration de systèmes d'information, a dès lors été entreprise un peu partout.

Mais, aujourd'hui, les techniques de l'information vivent un tournant. La croissance exponentielle de l'Internet a provoqué, dès le début des années 90, le foisonnement quasi anarchique de flux communicationnels. Submergées par la masse informationnelle, les organisations tentent de réagir, mais les outils classiques de gestion de l'information et de la communication révèlent leurs failles : les *mainframes* permettent une centralisation des données et des traitements, mais ils sont peu évolutifs et peu compatibles entre eux ; les systèmes client/serveur, quant à eux, paraissent plus modulables, mais sont moins aptes à centraliser de l'information distribuable.

Seules les technologies issues de l'Internet lui-même semblent proposer une solution prometteuse au problème qu'elles ont suscité. La technologie intranet est présentée par d'aucuns comme le cœur des systèmes d'information de demain : elle permettrait de centraliser l'information, tout en la distribuant de manière cohérente et adaptée.

L'Institut d'Informatique des Facultés Notre-Dame de la Paix s'intéresse, lui aussi, à cette nouvelle technologie : un intranet ne permettrait-il pas d'instaurer une communication plus efficace au sein des étudiants, des professeurs, des chercheurs et du personnel administratif ? Dans ce mémoire, nous nous sommes penché sur cette question. Plus précisément, nous avons tenté de commencer un intranet adapté à la situation concrète de l'Institut d'Informatique. Nous avons procédé en trois étapes : nous avons d'abord étudié les principes de base d'un intranet, puis nous avons analysé l'existant de l'Institut et, enfin, nous avons créé un prototype d'outil interactif.

Ainsi, dans notre premier chapitre, nous avons tout d'abord essayé de cerner les tenants et les aboutissants du concept d'intranet. Ce dernier nous est apparu comme « un outil de communication, basé sur les technologies de l'Internet, consacré à une organisation et à ses membres, pour offrir à ces derniers les ressources informationnelles, matérielles et logicielles de l'organisation, afin qu'ils les exploitent et les enrichissent, avec plus d'efficacité, de souplesse, de dynamisme et de facilité ».

Muni de cette définition, nous avons alors abordé l'infrastructure qu'un tel outil requiert : le matériel est celui d'un réseau normal, l'ensemble logiciel tient essentiellement dans un navigateur et les protocoles proviennent directement de l'Internet. Sur cette infrastructure, l'intranet doit être construit, à la fois sur le poste client et sur le serveur, au moyen de diverses méthodes : CGI, contrôles ActiveX, applets ou servlets Java… en accordant une vigilance particulière à la sécurité. Cette construction doit prendre corps dans une architecture, qui associe généralement un serveur de données, un serveur applicatif et un client, et qui utilise un intermédiaire entre le client et le(s) serveur(s) (un *middleware*).

Elaboré selon ces principes, un intranet peut alors rendre divers services à l'organisation, en termes de distribution d'informations et d'applications interactives. Des conséquences organisationnelles indirectes peuvent même être observées : l'intranet peut favoriser un fonctionnement plus souple, qui s'articule autour de la décentralisation de la prise de décisions et de la création de points de contrôle.

Désormais éclairé sur les principes de base d'un intranet, nous nous sommes alors concentré sur la possibilité d'en installer un à l'Institut. Notre deuxième chapitre a ainsi tout d'abord exposé les pré-requis indispensables à la mise en œuvre d'un intranet et les éléments clés d'un projet de ce type : nous avons retenu qu'un intranet devait s'appuyer sur une standardisation des composants du système d'information, une centralisation des données vers un serveur et une priorité conférée à l'utilisateur, ainsi que sur une préparation soignée. Nous avons donc procédé à une analyse de l'existant, afin de vérifier la présence des pré-requis à l'intranet. Ensuite, à partir d'un examen du bourgeon d'intranet d'ores et déjà implanté à l'Institut, nous avons dégagé les principes organisationnels, techniques et fonctionnels qui doivent déterminer notre projet. Il nous a semblé, en fonction du résultat de ces réflexions, que l'Institut pouvait effectivement utiliser les éléments existants, pour les prolonger dans un intranet, à proprement parler.

Ainsi, avons-nous effectué une étude des besoins de l'Institut, en termes de communications électroniques : nous les avons rassemblés selon leur portée administrative, pédagogique et scientifique, et nous avons donné pour chacun une brève description, ainsi que les contraintes organisationnelles et techniques qui nous apparaissaient d'emblée. Pour évaluer chaque besoin en fonction de son utilité et de ses risques, nous avons forgé nos propres critères et nous avons effectué un *scoring*. Nous avons alors dégagé les projets les plus intéressants et les plus facilement réalisables, de ceux qui présentaient moins d'utilité ou plus de risques. C'est le projet des valves électroniques qui a alors montré le meilleur rapport utilité/risque.

Pour atteindre le deuxième objectif poursuivi par ce mémoire, à savoir la mise en place concrète d'un incrément d'intranet, nous avons alors décidé de concevoir et de créer ces valves électroniques : notre troisième chapitre y est consacré. Bien sûr, nous n'avons pas la prétention de réaliser un intranet achevé : un intranet ne cesse de se développer et se remodeler, et les applications qui le composent doivent s'inscrire dans cette logique inchoative. Non seulement les valves électroniques ne constituent qu'un point de départ, mais de plus, notre résultat n'est qu'un prototype destiné à évoluer.

Afin que ce dernier soit néanmoins solide et convainquant, nous avons commencé par approfondir notre compréhension du besoin éprouvé à l'Institut pour des valves électroniques. Ainsi, il nous est apparu que la migration électronique des valves était soumise à des contraintes fonctionnelles, techniques et organisationnelles assez importantes. Au niveau fonctionnel, la solution électronique se doit d'apporter des avantages sur le système physique. Au niveau technique, les choix concrets d'implémentation doivent garantir une facilité de maintenance, une efficacité de consultation et une certaine sécurité des valves. Au niveau organisationnel, nous avons souligné la nécessité de ne pas accroître la charge de travail des membres de l'Institut et nous avons réfléchi sur les réticences à distribuer le pouvoir d'ajout et de suppression d'avis aux valves. De l'ensemble de ces contraintes, nous avons conclu que notre application devait comporter un système novateur de nettoyage automatique des avis « périmés » et des avis « en attente », qu'elle serait formée de scripts Unix écrits en Bash, qu'elle génerait des fichiers HTML, qu'elle profiterait des procédures de sécurité déjà en

vigueur, qu’elle ne nécessiterait qu’un *Webmaster* technique et qu’elle pourrait permettre à l’ensemble du staff, mais pas aux étudiants, de déposer et de supprimer des avis. Une fois ces principes fondamentaux posés, nous nous sommes alors penché sur la conception proprement dite de notre application.

La réalisation d’un schéma entité-association nous a tout d’abord éclairé sur les diverses entités impliquées dans les valves électroniques, et sur les relations qu’elles entretiennent entre elles. Nous avons ensuite élaboré une architecture, au moyen d’une méthode personnelle qui combine deux types d’architecture. Ainsi, une architecture orientée objet nous a renseigné sur les traitements à prévoir pour les entités centrales du schéma entité-association : les méthodes des objets valves, avis et erreur ont ainsi été portées au jour. Ces méthodes ont alors été transformées en des modules, au sein d’une architecture *top-down*, conformément aux exigences de la programmation impérative requise par les scripts Bash. A partir de trois modules généraux, tous les modules intermédiaires et finaux ont pu être déduits. Concrètement, ils ont été implantés dans quatre scripts : l’ajout d’un avis à au moins une valve, la consultation d’une valve pour la suppression d’un avis, la suppression d’un avis et la mise à jour des valves. Il nous a semblé préférable de concevoir ces scripts de façon indépendante et unitaire, plutôt que d’employer des boîtes à outils communes : un système de réplication des fonctions dans les scripts permet d’assurer une cohérence générale et une maintenance relativement aisée.

Puisque notre architecture nous a montré quelles fonctions créer, nous devions encore spécifier chaque module (en fonction de ses pré-conditions, de ses post-conditions, des modules qu’il utilise et d’une description plus libre), et en rédiger le code exécutable. La spécification des modules est détaillée dans la première annexe de ce mémoire : nous n’avons pas jugé opportun de la traiter comme telle dans le texte du mémoire. De même, nous n’avons pas abordé les divers problèmes concrets que la rédaction du code a suscités, et la manière dont nous les avons résolus : ce sont souvent des techniques liées au langage et à l’environnement employés, ou des difficultés purement algorithmiques. Pour le lecteur intéressé, le détail du code est reproduit dans les annexes : on trouvera, dans la deuxième annexe, le texte de chacun de nos quatre scripts, écrit en Bash, et dans la troisième annexe, les fonctions de validation des données internes au formulaire d’ajout d’un avis, écrites en JavaScript,

Pour que notre application soit achevée, nous avons enfin élaboré l’interface homme-machine, à savoir les valves en tant que pages Web consultables, le formulaire d’ajout d’un avis et le formulaire de suppression d’un avis. Pour ce faire, nous avons identifié les contraintes qui pèsent sur les fonctionnalités interactives afin de les traduire dans une présentation adéquate et conviviale.

Comme les composants d’un intranet sont destinés à évoluer constamment, il nous a semblé intéressant de terminer notre troisième chapitre en tentant de prévoir, d’ores et déjà, quelques aménagements, dont nos valves électroniques pourraient bénéficier dans l’avenir, si le besoin s’en faisait sentir. Ainsi, nous avons envisagé de changer le *Webmaster*, de diminuer le temps de réponse d’une demande d’ajout ou de suppression d’avis, de mieux gérer l’indisponibilité des valves, d’ajouter une nouvelle valve et d’introduire un degré d’importance dans chaque avis. Bien sûr, il ne s’agit ici que d’indications, et non de marches à suivre systématiques.

Avec ces valves électroniques, dont le fonctionnement peut être testé à l'URL suivante : « http://www.info.fundp.ac.be/~ffilee/intranet.html », nous avons donc posé la première pierre d'un édifice en perpétuelle construction. Certes, les membres du staff disposaient déjà d'applications interactives, pour la réservation de locaux ou de matériaux, mais celles-ci ne concernaient pas l'ensemble de l'Institut. Or, nos valves se situent au cœur même du fonctionnement académique : professeurs, chercheurs, étudiants et personnel administratif les utilisent couramment.

Cette utilisation importante des valves doit nous amener à relativiser notre travail, autant qu'à admettre son rôle. D'une part, en effet, notre application risque bien vite d'apparaître comme incomplète ou insuffisante : le comportement d'un programme est toujours différent en situation réelle, de nouveaux besoins peuvent toujours naître d'une confrontation avec un logiciel… Le résultat de nos travaux relève plutôt d'un prototype, destiné à être malmené et adapté.

D'autre part, malgré cette valeur relative, notre prototype porte néanmoins une responsabilité importante. En effet, le succès ou l'échec des valves électroniques ne se réduit pas à un enjeu ponctuel, lié à un programme particulier. A travers notre prototype, il est vraisemblable que les utilisateurs auront tendance à, d'ores et déjà, se forger une opinion sur l'intranet lui-même : en cas d'insatisfaction face aux valves, les utilisateurs risquent d'éprouver de la réticence face au jeune projet intranet ; à l'inverse, s'ils sont contents, ils verront d'un œil positif la suite du projet et pourront peut-être même s'y investir. Notre petite application va sans doute servir de témoin.

Dès lors, il est fondamental que d'autres, après nous, se chargent de sa maintenance et de son évolution.

Bibliographie

Nous avons utilisé directement les références suivantes :

ALIN Fr., LAFONT D., MACARY J.-Fr., *Le projet intranet. De l'analyse des besoins de l'entreprise à la mise en œuvre de solutions*, Paris, Eyrolles, Coll. Fi System, 1998.

BERNARD R., *The Corporate Intranet : Create and Manage an Internal Web for Your Organization, John Wiley & Sons*, 1996.

BICKEL R., *Building Intranets. Internal Webs give companies a new solution to an old problem*, Reprinted from *Internet World magazine*, vol. 7 n° 3, Copyright © 1996 Mecklermedia Corporation, cf. www.internetworld.com/print/monthly/1996/03/intranet.html

BODART Fr., PIGNEUR Y., *Conception assistée des systèmes d'information. Méthode, modèles, outils*, Paris-Milan-Barcelone-Bonn, Masson, Coll. Méthodes Informatiques et Pratiques des Systèmes, 1993, 2ème édition, 2ème tirage.

BRENY A.-M., *N.U.R. : Proposition pour un calendrier de mise en place des pages administratives sur le Web*, Namur, F.U.N.D.P. - Institut d'Informatique, document interne, avril 1996.

BRENY A.-M., GOBERT X., GOOSENS P., *N.U.R. : Proposition de réalisations pour un support administratif*, Namur, F.U.N.D.P. - Institut d'Informatique, document interne, octobre 1996.

CASSELBERRY R., et al., *Running a Perfect Intranet*, QueCorporation, 1996, HTML conversion by M/s. LeafWriters (India) Pvt. Ltd., cf. http://www.mcp.com/849139200/0-7897/0-7897-0823-X/index.htm

DANCEL A., *Programmation avancée sous Unix*, Ecole Nationale de l'Aviation Civile, juillet 1994.

DOMINIQUE C., *Ressources CGI*, Mars 1998, cf. http://www.web developpeur.com/Conception/ressourcesCGI.html

ESPLIN K., *8 important issues to consider before building an intranet, What must you know about your intranet's infrastructure and staffing to be successful? These guidelines will help you through the morass of intranet planning. Plus Sun and Lockheed Martin tell how they constructed their massive intranets*, Mars 1997, cf. www.sunworld.com/swol-03-1997/swol-03-intranet.html

GAUTIER R., *Qu'est-ce qu'un intranet ?*, Communication Jean Lalonde 1999, cf. www.cjl.qc.ca/batisseurs/intranet.htm

GOOSSENS P., *Nouvelles Utilisations Réseaux*, Namur, F.U.N.D.P. - Institut d'Informatique, document interne, [sans date].

HEKMAN J. P., *Linux in a Nutshell. Guide de référence*, Paris, O'Reilly, 1997, trad. NADEAU A. et VANSTEENE J.-M.

KHASNABISH B., SARACCO R., *Intranets : Technologies, Services and Management*, in *IEEE Communications Magazine*, October 1997.

KOCH S., *Introduction to JavaScript*, 1996, cf. http://rummelplatz.uni-mannheim.de/~skoch/js/script.htm, trad. fr. ORTUNIO L., cf. http://wwwusers.imaginet.fr/~ortunio/javascript/script.htm

LACROIX Fr., BOG D., *Le langage AWK*, Grenoble, ENSIMAG, Année Spéciale Informatique, 1992.

OLIVER D., HOLZSCHALG M., *HTML 4*, Ed. Simon & Schuster Macmillan, Coll. Le tout en poche, 1998, trad. EBERHARDT C. et CAMPRILLO V.

PARKER M. M., BENSON R. J., TRAINOR H. E., *Information Economics-Linking Business Performance to Information Technology*, Prentice Hall, 1988

PLANQUE F., *Le CGI - Common Gateway Interface*, Août 1997, cf. http://www.webdeveloppeur.com/Conception/introductionCGI.html

SNELL N., *Internet*, Ed. Simon & Schuster Macmillan, Coll. Le tout en poche, 1998, trad. PROUT D.

STUHRMANN C., *Les meilleurs sites Internet francophones 1999*, Micro Application, Coll. PC Poche, 1999, 2ème édition.

TANENBAUM Andrew, *Réseaux*, Paris/London, Inter-Editions/Prentice-Hall, 1997, 3ème édition, trad. française par HERNANDEZ J.-A. et JOLY R.., titre original : *Computer Networks*.

TELLEEN St. L., *IntraNet Methodology. Concepts and Rationale*, Amdahl Corporation, Sunnyvale, California, USA, 1996, cf. http://www.amdahl.com/doc/products/bsg/intra/concept.htm

TELLEEN St. L., *Intranets and Adaptive Innovation: The move from control to coordination in today's organizations*, Amdahl Corporation, Sunnyvale, California, USA, 1996, cf. http://www.amdahl.com/doc/products/bsg/intra/adapt.htm

TELLEEN St. L., *The IntraNet Architecture: Managing information in the new paradigm*, Amdahl Corporation, Sunnyvale, California, USA, 1996, cf. http://www.amdahl.com/doc/products/bsg/intra/infra.htm

XXX, *10 Intranet Myths*, Smart Infotech Systems Ltd., 1997, cf. www.intrack.com/intranet/intmyth10.html

XXX, *JavaScript Guide*, © Netscape Communications Corporation 1996, Mountain View (U.S.A.).

XXX, *The Intranet. Implementation of Internet And Web Technologies*, Organizational Information Systems, Hummingbird Communications Ltd., 1996, cf. www.hummingbird.com/whites/intranet.html

WIELSCH M., *Linux*, Micro Application, Coll. PC-Poche, 1996.

Nous avons également consulté les sites Web suivants :

- pour la définition d'un intranet :
  http://www.microsoft.com/intranet/
  http://www.microsoft.com/office/intranet/
  http://www.microsoft.com/france/intranet/Index.htm
  http://www.radian.fr/intranet.html

- pour la réalisation d'un intranet :
  http://www.sun.com/products-n-solutions/hw/servers/netra/netra_i/index.html
  http://www.javasoft.com

- pour des revues :
  http://www.intranetjournal.com/
  http://www.innergy.com/index.html
  http://www.intrack.com/intranet/

- pour des études de cas
  http://home.netscape.com/comprod/at_work/customer_profiles/lilly.html
  http://home.netscape.com/comprod/at_work/customer_profiles/john_deere.html
  http://www.cio.com/WebMaster/feature_nov95.html

Annexe 1. Spécifications des modules de l'application

Bisext

Pré-conditions

La fonction prend un argument : un nombre entier.

Post-conditions

Elle renvoie :

- 1 si l'argument correspond à une année bissextile
- 0 sinon.

Utilise

Ce module n'en utilise aucun autre (module terminal)

Description

La fonction vérifie si une année est bissextile. Elle l'est si et seulement si elle est divisible par quatre, sauf si elle est divisible par 100 et pas par 400. Logiquement, cela s'exprime comme suit :

Bisext = (Annee Mod 4 = 0) **et** ((Annee Mod 100 ≠ 0) **ou** (Annee Mod 400 = 0))

où Annee est un nombre entier quelconque

CompareDate

Pré-conditions

La fonction prend six arguments : six nombres entiers.

Post-conditions

Sachant que Date1 = (argument1, argument2, argument3) et Date2 = (argument4, argument5, argument6), le module renvoie :

- 1 si Date1 < Date2,
- 0 si Date1 > Date2,
- 2 si Date1 = Date2

Utilise

Ce module n'en utilise aucun autre (module terminal)

Description

La fonction prend les trois premiers arguments comme étant le jour, le mois et l'année d'une première date et les trois arguments suivants comme étant le jour, le mois et l'année d'une deuxième date. Elle compare ces dates pour les évaluer l'une par rapport à l'autre.

ControleApresVacuiteValve

Pré-conditions

La fonction prend trois arguments :

- le nom générique de la valve dont il faut contrôler la vacuité (ValveStaff, Valve1LM, Valve2LM, Valve3M),
- le type de cette valve (Reel, Virtuel)
- le nom complet du fichier de cette valve (de type *.html).

Post-conditions

La fonction veille à ce qu'une valve qui ne contient plus d'avis soit marquée comme étant une valve vide. Elle retourne toujours 1.

Utilise

Ce module en utilise deux autres :

- EstVideValve
- CreerValveVide

Description

La fonction teste si le troisième argument correspond à un fichier vide. Si c'est le cas, elle crée une valve vide. Sinon, elle ne fait rien.

ControleModifValve

Pré-conditions

La fonction prend deux arguments :

- le nom d'un fichier de valves (*.html)
- le nom d'un fichier de propriétés de valves (*.Prop).

Post-conditions

Elle renvoie :

- 1 si les données contenues dans le fichier désigné par le deuxième argument correspondent aux données actuelles du fichier désigné par le premier argument
- 0 si les données sont différentes
- 2 si le fichier des propriétés n'existe pas.

Utilise

Ce module utilise le module ExisteFichier.

Description

Le fichier de propriétés a été construit par le script ConsultSupprValve.cgi en effectuant la commande Unix « ls » sur le fichier des valves et en y ajoutant le numéro IP de la machine hôte (variable $REMOTE_HOST). Cette fonction compare les propriétés du fichier des valves lorsqu'il a été consulté sous la forme du formulaire de suppression et les propriétés actuelles du fichier des valves (en effectuant un « ls » et en y ajoutant la variable $REMOTE_HOST).

CreerValve

Pré-conditions

La fonction prend trois paramètres :

- le nom générique de la valve (ValveStaff, Valve1LM, Valve2LM, Valve3M)
- le type de valve à créer (Virtuel, Reel)
- le nom du fichier de la valve à créer (*.html).

Les variables globales JourSysteme, MoisSysteme, AnneeSysteme, qui donnent la date du jour, sont requises.

Post-conditions

Son résultat est la création de la valve correspondant aux exigences des deux premiers arguments sous le nom spécifié par le troisième argument. La fonction renvoie toujours 1.

Utilise

Ce module n'en utilise aucun autre.

Description

La fonction crée une valve destinée à recevoir un ou plusieurs avis : il n'y donc pas de marque de valve vide insérée ici.

CreerValveVide

Pré-conditions

La fonction prend trois paramètres :

- le nom générique de la valve (ValveStaff, Valve1LM, Valve2LM, Valve3M)
- le type de valve vide à créer (Virtuel, Reel)
- le nom du fichier de la valve vide à créer (*.html).

Les variables globales JourSysteme, MoisSysteme, AnneeSysteme, qui donnent la date du jour, sont requises.

Post-conditions

Son résultat est la création de la valve vide correspondant aux exigences des deux premiers arguments sous le nom spécifié par le troisième argument. La fonction renvoie toujours 1.

Utilise

Ce module n'en utilise aucun autre.

Description

La fonction crée une valve vide. Une marque spécifique de valve vide est insérée ici en lieu et place d'un avis.

DecoderDonneesAjoutAvis

Pré-conditions

La fonction requiert, sous forme de variables globales initialisées, les variables qui sont contenues dans le formulaire d'ajout d'un avis. Voici ces variables avec leur initialisation :

- FORM_TitreAvis = ''
- FORM_ContenuAvis = ''
- FORM_DeJour = ''
- FORM_DeMois = ''
- FORM_DeAnnee = ''
- FORM_AuJour = ''
- FORM_AuMois = ''
- FORM_AuAnnee = ''
- FORM_OkStaff = 'OFF'
- FORM_Ok1LM = 'OFF'
- FORM_Ok2LM = 'OFF'
- FORM_Ok3M = 'OFF'
- Auteur_Avis = ''

Post-conditions

La fonction remplace le contenu de ces variables globales par les données introduites dans le formulaire. La variable Auteur_Avis reçoit le contenu de la variable $REMOTE_HOST.

Utilise

Ce module appelle le module /opt/blanc/comms/httpd/bin/cgiparse, qui assume la responsabilité du décodage le formulaire. Ce module est écrit en C ; c'est un programme extérieure à notre application.

Description

Il s'agit simplement d'appeler le service de *parsing* de l'Institut. Depuis le formulaire, les valeurs des variables sont envoyées par HTTP sur le standard input. Le script cgiparse décode cet input et génère une suite d'instructions de la forme « FORM_variable = valeur_decodée ». Nous pouvons ensuite récupérer ces variables dans notre script.

DecoderDonneesSuppressionAvis

Pré-conditions

La fonction requiert, sous forme de variables globales initialisées, les variables qui sont contenues dans le formulaire de suppression d'un avis. Voici ces variables avec leur initialisation :

- FORM_NomValve = ''
- FORM_TypeValve = ''
- FORM_NumAvisSuppr = ''

Post-conditions

La fonction remplace le contenu de ces variables globales par les données introduites dans le formulaire.

Utilise

Ce module appelle le module /opt/blanc/comms/httpd/bin/cgiparse –form, tout comme le module précédent.

Description

Voir module précédent.

DeplacerAvisActivable

Pré-conditions

Cette fonction requiert des variables globales

- ListeAvisActivable : la suite des numéros d'avis à déplacer
- NomFichierVirtuel : le nom du fichier des valves virtuelles concernées
- NomValveInterm : le nom du fichier temporaire où les avis activables seront déplacés
- NomFIntermVirtuel : le nom du fichier temporaire où les valves virtuelles, sans les avis activables, seront copiées.

Post-conditions

La fonction s'occupe de déplacer les avis activables des valves virtuelles (variable NomFichierVirtuel) vers un fichier temporaire (NomValveInterm), et de copier ce qui reste des valves virtuelles dans un autre fichier temporaire (NomFIntermVirtuel). Elle renvoie toujours 1.

Utilise

Ce module n'en utilise aucun autre.

Description

La fonction parcourt séquentiellement le fichier des avis virtuel : au fur et à mesure, elle copie les avis activables dans un fichier et les avis non-activables dans un autre.

EmpecherSimultaneite

Pré-conditions

Le nom du fichier qui permet de fermer l'accès aux valves doit être « LockValve ».

Post-conditions

La fonction renvoie

- 0 si le fichier LockValve existe déjà
- 1 si le fichier LockValve n'existe pas et est donc créé.

Utilise

Ce module utilise le module ExisteFichier.

Description

Pour empêcher tout traitement simultané sur les valves, cette fonction vérifie l'existence du fichier LockValve. Si ce fichier n'existe pas, la fonction le crée et s'approprie ainsi le droit de modification des valves.

EmpecherSimultaneiteCons

Pré-conditions

Le nom du fichier qui permet de fermer l'accès aux valves doit être « LockValve ».

Post-conditions

La fonction renvoie

- 0 si le fichier LockValve existe déjà
- 1 si le fichier LockValve n'existe pas.

Utilise

Ce module utilise le module ExisteFichier.

Description

Pour empêcher d'afficher une valve qui est en train d'être modifiée, cette fonction vérifie simplement l'existence du fichier LockValve. Ce module, contrairement au précédent, ne crée pas le fichier LockValve, car la consultation d'une valve ne la modifie pas !

EmpecherSimultaneiteMaj

Pré-conditions

Le nom du fichier qui permet de fermer l'accès aux valves doit être « LockValve ».

La fonction requiert les variables globales suivantes, qui sont des constantes :

- NomScript : le nom du script général (ici, MiseAJourValve)
- Destinataire : le nom du *Webmaster* à avertir en cas de problème
- Sujet : le sujet du *mail* envoyé au *Webmaster*.

Post-conditions

La fonction renvoie

- 0 si le fichier LockValve existe et a été modifié durant les trois dernières minutes (les valves sont en modification). La fonction programme l'exécution du script dix minutes plus tard
- 1 si le fichier LockValve n'existe pas, ou s'il n'a pas été modifié durant les trois dernières minutes (les valves ne sont pas en modification).

Utilise

Ce module utilise les modules suivants :

- ExisteFichier
- EnvoiMail.

Description

Cette fonction se veut plus complète que les précédentes. Elle cherche à empêcher tout travail simultané sur les valves, mais aussi que celles-ci soient bloquées par la présence permanente du fichier LockValve. Si le fichier LockValve existe, la fonction va attendre trois minutes et vérifier ensuite que ce fichier (s'il existe encore) a été modifié depuis. Un traitement sur les valves ne durera jamais plus de trois minutes. Si ce fichier a été modifié, c'est que les valves sont effectivement en train de subir divers traitements… la fonction retarde alors l'exécution du script de dix minutes. Si le fichier n'a pas été modifié durant les trois minutes, la fonction suppose qu'un script n'a pas correctement terminé son travail et n'a pas effacé le fichier LockValve : elle envoie alors un mail au *Webmaster* pour l'informer de cette anormalité. La fonction va ensuite se réapproprier le fichier pour fermer l'accès aux valves.

EnregProprietesValve

Pré-conditions

La fonction reçoit deux paramètres :

- le nom du fichier de la valve concernée (*.html)
- le nom du fichier des propriétés de la valve (*.Prop).

Post-conditions

La fonction provoque l'enregistrement des propriétés du fichier désigné par le premier paramètre, dans le fichier désigné par le second paramètre. Le numéro IP de la machine hôte est également enregistré dans ce deuxième fichier.

Utilise

Ce module n'en utilise aucun autre.

Description

Lorsqu'une valve est affichée dans le formulaire de consultation d'avis (c'est le script ConsultSupprValve qui entre en jeu), ce module doit permettre de garder une trace des propriétés du fichier consulté, ainsi que de l'identité de celui qui le consulte.

EnvoiMail

Pré-conditions

La fonction peut recevoir quatre arguments :

- le nom du destinataire du *mail* à envoyer (la constante Destinataire est spécialement prévue à cet effet)
- le sujet du *mail* à envoyer (la constante Sujet est spécialement prévue à cet effet)
- le type d'erreur ('ValveInexistante' ou 'PresenceLockValve')
- éventuellement le nom du fichier incriminé (pour 'ValveInexistante').

Post-conditions

Elle provoque l'envoi d'un courrier électronique à la personne donnée comme premier argument, dont le sujet est donné par le deuxième argument. Grâce au troisième argument, la fonction « crée » un texte correspondant au type d'erreur. Si cette dernière relevait du type 'ValveInexistante', le nom du fichier inexistant, donné comme quatrième argument, serait inséré dans le texte du *mail*. La fonction renvoie toujours 1.

Utilise

Ce module n'en utilise aucun autre.

Description

Deux types d'erreur peuvent nécessiter l'envoi d'un courriel au *Webmaster* : d'une part, la détection d'une valve vide par n'importe lequel de nos quatre scripts ; d'autre part, la présence inopportune du fichier LockValve sur le serveur, telle qu'elle a pu être diagnostiquée par le script MiseAJourValve, (c'est-à-dire par le EmpecherSimultaneiteMaj).

ErreurAjout

Pré-conditions

La fonction prend deux paramètres :

- le message à afficher sur le type d'erreur
- un conseil pour l'utilisateur.

Post-conditions

Elle provoque l'affichage d'une page HTML, dont le titre est « Résultat d'un ajout d'avis aux valves », et dont le contenu est donné par les deux paramètres.

Utilise

Ce module n'en utilise aucun autre.

Description

L'affichage HTML sera conforme à la présentation des valves et des formulaires.

ErreurSuppression

Pré-conditions

La fonction prend deux paramètres :

- le message à afficher sur le type d'erreur
- un conseil pour l'utilisateur.

Post-conditions

Elle provoque l'affichage d'une page HTML, dont le titre est « Résultat de la suppression d'un avis », et dont le contenu est donné par les deux paramètres.

Utilise

Ce module n'en utilise aucun autre.

Description

L'affichage HTML sera conforme à la présentation des valves et des formulaires.

ErreurValidation

Pré-conditions

La fonction prend un paramètre : le message à afficher sur le type d'invalidité des données.

Post-conditions

Elle provoque l'affichage d'une page HTML, dont le titre est « Résultat d'un ajout d'avis aux valves ». La première phrase est la suivante : « Votre demande n'a pas été acceptée ». Le deuxième message est donné par le paramètre. Puis une invitation à recommencer la saisie avec des données cohérentes conclut la page d'erreur.

Utilise

Ce module n'en utilise aucun autre.

Description

L'affichage HTML sera conforme à la présentation des valves et des formulaires.

EstVideValve

Pré-conditions

La fonction prend un paramètre : le nom du fichier de la valve (*.html).

Post-conditions

Elle renvoie :

- 0 si le fichier spécifié comme paramètre n'est pas vide
- 1 si ce fichier est vide

Utilise

Ce module n'en utilise aucun autre.

Description

Cette fonction pourrait fonctionner de plusieurs manières. Nous optons pour le parcours séquentiel du fichier spécifié jusqu'à une marque de début d'avis : une seule suffit à conclure que la valve n'est pas vide. Cette solution nous paraît plus efficace que de lancer une commande « grep ».

EtablirListeValve

Pré-conditions

La fonction requiert les variables globales suivantes, qui sont des constantes :

- ListeValvePossible : la liste des noms génériques de valves possibles (ValveStaff, Valve1LM, Valve2LM, Valve3M)
- Destinataire : le nom du *Webmaster* à avertir en cas de problème
- Sujet : le sujet du *mail* envoyé au *Webmaster.*

Elle requiert aussi une variable globale, initialisée de la manière suivante : ListeValveExistant = ''.

Post-conditions

Elle renvoie

- 0 si la liste des valves existantes est vide
- 1 si la liste des valves existantes n'est pas vide.

Utilise

Ce module utilise les modules suivants :

- ExisteFichier
- CreerValveVide
- EnvoiMail

Description

La fonction cherche des valves existantes. Pour chaque nom générique de valve repris dans la variable ListeValvePossible, la fonction recherche l'existence de la valve réelle et de la valve virtuelle correspondantes. Si la valve réelle ou la valve virtuelle est trouvée, la fonction ajoute le nom générique de cette valve à la variable ListeValveExistant. Pour toutes les valves inexistantes, la fonction crée une valve vide et en avertit le *Webmaster*. Une fois que la liste des valves possibles est épuisée, la fonction examine si la variable ListeValveExistant contient au moins un élément.

ExisteFichier

Pré-conditions

La fonction reçoit un paramètre : le nom du fichier.

Post-conditions

Elle renvoie

- 0 si le fichier n'existe pas
- 1 si le fichier existe.

Utilise

Ce module n'en utilise aucun autre.

Description

Cette fonction est extrêmement simple : la commande Unix « test », avec l'option « - e », lui donne d'emblée le résultat.

IdentifierTraitement

Pré-conditions

La fonction requiert trois variables globales

- NomFichierReel : le nom du fichier des valves réelles
- NomFichierVirtuel : le nom du fichier des valves virtuelles
- Traitement : qui doit être initialisée à vide.

Post-conditions

La fonction identifie le traitement de mise à jour à effectuer sur les valves spécifiées dans les variables globales NomFichierReel et NomFichierVirtuel. La variable globale Traitement reçoit une des valeurs suivantes :

- MAJActualise, si le traitement consiste seulement dans une actualisation,
- MAJPerime, si le traitement consiste seulement dans une péremption
- MAJTotal, si le traitement consiste à la fois en une actualisation et une péremption.

La fonction renvoie :

- 0 si aucun traitement ne doit être effectué
- 1 si un traitement doit être effectué.

Utilise

Ce module utilise le module EstVideValve.

Description

Pour identifier le traitement à effectuer étant donné le nom générique d'une valve, il suffit de tester si la valve réelle est vide ou si la valve virtuelle est vide :

- si les deux sont vides, aucun traitement n'est à effectuer ;
- si seule la valve réelle est vide, le traitement est une actualisation ;
- si seule la valve virtuelle est vide, le traitement est une péremption ;
- si les deux valves ne sont pas vides, le traitement est total.

InsererAvisValve

Pré-conditions

La fonction requiert les variables globales suivantes :

- NomFichier : le nom du fichier dans lequel il faut insérer l'avis
- NomFichInterm : le nom du fichier intermédiaire pour constituer le nouveau fichier
- TitreAvis : le titre de l'avis à insérer
- ContenuAvis : le corps de l'avis à insérer
- JourDepotAvis : le jour de dépôt de l'avis à insérer
- MoisDepotAvis : le mois de dépôt de l'avis à insérer
- AnneeDepotAvis : l'année de dépôt de l'avis à insérer
- JourPeremptionAvis : le jour de péremption de l'avis à insérer
- MoisPeremptionAvis : le mois de péremption de l'avis à insérer
- AnneePeremptionAvis : l'année de péremption de l'avis à insérer
- AuteurAvis : l'auteur de l'avis à insérer (numéro IP).

Post-conditions

La fonction provoque la création d'un fichier intermédiaire NomFichInterm, qui est le résultat de l'insertion de l'avis dans le fichier NomFichier. Ces deux fichiers ne sont ni effacés, ni renommés par la fonction.

Utilise

Ce module n'en utilise aucun autre.

Description

Pour insérer l'avis, on parcours le fichier des valves séquentiellement. On recopie chaque ligne dans le fichier intermédiaire, jusqu'à la marque de fin d'en-tête ; puis on copie les données de l'avis dans ce fichier intermédiaire ; et ensuite on continue le parcourt séquentiel du fichier en recopiant chaque ligne.

ListerAvisActivable

Pré-conditions

La fonction requiert les variables globales suivantes :

- ListeAvisActivable : qui doit être initialisée à vide
- NomFichierVirtuel : le nom du fichier de la valve virtuelle (*.html)

Les variables globales JourSysteme, MoisSysteme, AnneeSysteme, qui donnent la date du jour, sont aussi nécessaires.

Post-conditions

La variable ListeAvisActivable reçoit la liste des numéros des avis d'une valve virtuelle (spécifiée par la variable NomFichierVirtuel) dont la date de dépôt est antérieure ou égale à celle de ce jour.

Elle renvoie toujours 1.

Utilise

Ce module utilise le module CompareDate

Description

Pour établir la liste des avis activables, il suffit de parcourir le fichier spécifié séquentiellement. A chaque marque de date de dépôt d'avis, il suffit de capturer le jour, le mois et l'année indiqués, puis de comparer cette date avec celle du jour.

ListerAvisPerime

Pré-conditions

La fonction requiert les variables globales suivantes :

- ListeAvisPerime : qui doit être initialisée à vide
- NomFichierReel : le nom du fichier de la valve réelle (*.html)

Les variables globales JourSysteme, MoisSysteme, AnneeSysteme, qui donnent la date du jour, sont aussi nécessaires.

Post-conditions

La variable ListeAvisPerime reçoit la liste des numéros des avis d'une valve réelle (spécifiée par la variable NomFichierReel) dont la date de péremption est strictement postérieure à celle du jour.

Elle renvoie toujours 1.

Utilise

Ce module utilise le module CompareDate.

Description

Pour établir la liste des avis périmés, il suffit de parcourir le fichier spécifié séquentiellement. A chaque marque de date de péremption d'avis, il suffit de capturer le jour, le mois et l'année indiqués, puis de comparer cette date avec celle du jour.

MiseAJourValveReelle

Pré-conditions

La fonction requiert les variables globales suivantes :

- ListeAvisPerime : qui doit être initialisée à vide
- NomFichierReel : le nom du fichier de la valve réelle (*.html)
- NomFichReelInterm : le nom du fichier intermédiaire qui permettre de reconstituer les valves réelles
- NomValveInterm : le nom du fichier intermédiaire où sont les avis qui ont été déplacés de la valve virtuelle pour être insérés dans la valve réelle (actualisation)
- NomValve : le nom générique de la valve à traiter (ValveStaff, Valve1LM, Valve2LM, Valve3M)
- Traitement : le type de traitement qui doit être effectué (MAJActualise, MAJPerime, MAJTotal)

Post-conditions

Si le traitement est MAJActualise, la fonction crée une valve réelle (non vide) du même nom générique que la valve virtuelle existante. Dans tous les cas, la fonction crée un fichier temporaire (NomFReelInterm), qui est le résultat de l'insertion des avis à déplacer de la valve virtuelle vers la valve réelle (repris dans ValveInterm), et en même temps le résultat de l'effacement des avis périmés (dont la liste est dans ListeAvisPerime) au sein de la valve réelle.

Utilise

Ce module utilise le module CreerValve.

Description

Cette fonctionnalité veut mettre à jour une valve réelle en effectuant une seule lecture séquentielle de ce fichier. Le module va agir différemment selon le type de traitement à effectuer. Dans le cas d'une péremption, il s'agira simplement de copier le fichier des valves réelles vers un fichier temporaire, en omettant les avis périmés spécifiés dans ListeAvisPerime. Dans le cas d'une activation, il s'agira simplement de créer une valve réelle et d'y insérer les avis repris dans le fichier intermédiaire que le module DeplacerAvisActivable a créé. Dans le cas d'une mise à jour totale, il s'agira de gérer trois fichiers : le fichier temporaire contenant les avis à activer, le fichier des valves réelles dont il faut supprimer les avis périmés et le fichier temporaire qui va contenir le résultat ; tout d'abord, on insérera les avis activables, puis on copiera les valves réelles sans les avis périmés.

PublicationSuppressionHTML

Pré-conditions

La fonction prend un argument : le nom d'un fichier de valve (*.html).

Post-conditions

Elle provoque l'affichage, dans le navigateur, du fichier spécifié comme paramètre sous la forme d'un formulaire HTML où l'utilisateur peut choisir un avis à supprimer. Si la valve est vide, le module doit avertir l'utilisateur par un message adéquat.

Utilise

Ce module n'en utilise aucun autre.

Description

Il suffit de parcourir séquentiellement le fichier HTML spécifié, d'introduire de nouvelles balises dans certaines lignes et d'envoyer chaque ligne au navigateur.

SupprimerUnAvis

Pré-conditions

La fonction requiert les deux variables globales suivantes :

- NomFichier : le nom du fichier où il faut supprimer un avis
- NumAvisSuppr : le numéro de l'avis à supprimer dans ce fichier.

Post-conditions

Elle provoque la suppression, dans le fichier spécifié dans NomFichier, de l'avis dont le numéro est NumAvisSuppr.

Utilise

Ce module n'en utilise aucun autre.

Description

Pour supprimer un avis, cette fonction crée un fichier intermédiaire. Puis elle parcourt séquentiellement le fichier de la valve spécifiée et copie tous les avis sauf celui qui correspond au numéro à supprimer. Ensuite, la fonction copie le fichier temporaire sur l'ancien fichier.

TraiterAvis

Pré-conditions

La fonction requiert les variables globales suivantes :

- Dest : le nom générique de la valve dans laquelle un avis doit être ajouté
- TypeAvis : le type de l'avis à ajouter à une valve (Reel ou Virtuel). Cette variable donne aussi le type de valve dans laquelle l'avis doit être ajouté.

Elle nécessite aussi les constantes suivantes :

- Destinataire : le nom du Webmaster à avertir en cas de problème
- Sujet : le sujet du mail envoyé au *Webmaster.*

Post-conditions

Cette fonction provoque l'insertion d'un avis dans une valve, même si le fichier de celle-ci n'existe pas.

Elle renvoie toujours 1.

Utilise

Ce module utilise les modules suivants :

- CreerValve
- EnvoiMail
- ExisteFichier
- InsererAvisValve.

Description

Cette fonction doit d'abord s'assurer de l'existence du fichier des valves (formé par la concaténation des variables Dest, TypeAvis et de l'extension html : « Dest TypeAvis .html »), quitte à le créer elle-même. Ensuite, elle se contente d'appeler la fonction InsererAvisValve.

TraiterValve

Pré-conditions

La fonction requiert la variable globale :

- NomValve : le nom générique de la valve à mettre à jour

Elle nécessite aussi les constantes suivantes :

- Destinataire : le nom du Webmaster à avertir en cas de problème
- Sujet : le sujet du mail envoyé au *Webmaster.*

Post-conditions

La fonction se charge, étant donné un nom générique de valve, d’effectuer la mise à jour de la valve réelle et/ou de la valve virtuelle.

Utilise

Ce module utilise les modules suivants :

- IdentifierTraitement
- ListerAvisPerime
- ListerAvisActivable
- DeplacerAvisActivable
- MiseAJourValveReelle
- ControleApresVacuiteValve

Description

Tout d’abord, la fonction identifie le traitement à effectuer sur la valve réelle et/ou la valve virtuelle. Elle procède ensuite, selon le traitement, au relevé des avis périmés de la valve réelle, au relevé des avis activables dans la valve virtuelle, au déplacement de ces derniers dans un fichier temporaire et à la mise à jour de la valve réelle en fonction de ces données. Une fois toutes ces opérations réalisées, la fonction vérifie la cohérence des fichiers des valves, en s’assurant que les valves vides contiennent une marque explicite de valve vide.

TyperAvis

Pré-conditions

La fonction requiert les variables globales suivantes :

- JourDepotAvis : le jour de dépôt d’un avis
- MoisDepotAvis : le mois de dépôt d’un avis
- AnneeDepotAvis : l’année de dépôt d’un avis
- TypeAvis : qui doit être initialisée à vide.

Les variables globales JourSysteme, MoisSysteme, AnneeSysteme, qui donnent la date du jour, sont aussi requises.

Post-conditions

La variable TypeAvis reçoit la valeur ‘Virtuel’ ou ‘Reel’.

Utilise

Ce module utilise le module CompareDate.

Description

En comparant la date du jour à celle du dépôt de l’avis, cette fonction sait identifier si l’avis doit être ajouté dans une valve de type réelle ou dans une valve de type virtuelle.

ValideDate

Pré-conditions

La fonction prend trois paramètres qui doivent être des nombres entiers :

- le jour
- le mois
- l’année

Les variables globales JourSysteme, MoisSysteme, AnneeSysteme, qui donnent la date du jour, sont aussi requises.

Post-conditions

Elle renvoie :

- 1 si les trois paramètres forment une date correcte et postérieure ou égale à la date de ce jour
- 0 si les trois paramètres forment une date incorrecte
- 2 si la date formée est correcte, mais est antérieure à la date de ce jour.

Utilise

Ce module utilise les modules suivants :

- Bisext
- CompareDate.

Description

Cette fonction doit vérifier toutes les contraintes inhérentes à une date : un jour doit se situer entre 1 et 31, un mois doit se situer entre 1 et 12 et une année doit se situer entre 1999 et 2050 ; pour certains mois, le jour ne peut dépasser 30 ; pour le mois de février, si l'année est bissextile, le jour ne peut dépasser 29, sinon le jour ne peut dépasser 28. Enfin, si la date est validée, la fonction la comparera à la date de ce jour pour vérifier si elle ne lui est pas antérieure.

ValiderDonneesAjoutAvis

Pré-conditions

La fonction requiert les variables globales contenues dans le formulaire d'ajout d'un avis et qui ont été décodées par la fonction DecoderDonneesAjoutAvis : FORM_TitreAvis, FORM_ContenuAvis, FORM_DeJour, FORM_DeMois, FORM_DeAnnee, FORM_AuJour, FORM_AuMois, FORM_AuAnnee, FORM_OkStaff, FORM_Ok1LM, FORM_Ok2LM, FORM_Ok3M, ainsi que Auteur_Avis.

La fonction nécessite aussi les variables globales suivantes, avec leur initialisation respective :

- TitreAvis = ''
- ContenuAvis = ''
- JourDepotAvis = 0
- MoisDepotAvis = 0
- AnneeDepotAvis = 0
- JourPeremptionAvis = 0
- MoisPeremptionAvis = 0
- AnneePeremptionAvis = 0
- DestinataireAvis = 0

Les variables globales JourSysteme, MoisSysteme, AnneeSysteme, qui donnent la date du jour, sont aussi requises.

Post-conditions

Cette fonction transforme les variables telles qu'elles sont décodées du formulaire en variables valides. Dans le cas d'une incohérence des données, la fonction envoie un message d'erreur à l'utilisateur. TitreAvis reçoit le titre de l'avis, s'il n'est pas vide ; ContenuAvis reçoit le contenu de l'avis, s'il n'est pas vide ; JourDepotAvis, MoisDepotAvis, AnneeDepotAvis, JourPeremptionAvis, MoisPeremptionAvis, AnneePeremptionAvis constituent les dates de dépôt et de péremption de l'avis s'ils forment des dates valides, postérieures à la date de ce jour, et si la date de dépôt est strictement antérieure à celle de péremption ; DestinataireAvis reçoit la liste des noms

génériques des valves concernées par l'avis (ValveStaff, Valve1LM, Valve2LM, Valve3M).

Utilise

Ce module utilise les modules suivants :

- ValideDate
- ErreurValidation

Description

La fonction vérifie une à une les contraintes d'intégrité des données d'un avis à ajouter à des valves et transforme, au fur et à mesure, les variables reçues du formulaire en variables internes à l'application.

ValiderDonneesSupprAvis

Pré-conditions

La fonction requiert les variables globales contenues dans le formulaire d'ajout d'un avis et qui ont été décodées par la fonction DecoderDonneesSupprAvis : FORM_NomValve, FORM_TypeValve, FORM_NumAvisSuppr.

La fonction nécessite aussi les variables globales suivantes, avec leur initialisation respective :

- NomValve = ''
- TypeValve = ''
- NumAvisSuppr = 0

Les constantes suivantes sont également nécessaires :

- ListeValvePossible : la liste des noms génériques des valves possibles
- TypeValve : la liste des types de valves possibles.

Post-conditions

Cette fonction transforme les variables telles qu'elles sont décodées du formulaire en variables valides. NomValve reçoit le nom générique de la valve dans laquelle supprimer un avis, si ce nom existe dans la liste des valves possibles ; TypeValve reçoit le type de cette valve si ce type existe dans la liste des types possibles ; NumAvisSuppr reçoit le numéro de l'avis à supprimer, s'il est compris entre 1 et 99999.

La fonction renvoie :

- 0 si une erreur de validité des données a été détectée
- 1 si toutes les données apparaissent comme valides.

Utilise

Ce module n'en utilise aucun autre.

Description

La fonction vérifie l'intégrité des données nécessaires à la suppression d'un avis au sein d'une valve donnée.

Page blanche

Annexe 2. Code des scripts

Annexe 2.1. Le script AjoutAvisValve

```
#!/opt/blanc/gnu/bin/bash
# ++++++++++++++++++++++++++++++++++++++++++++++++
# APPLICATION DE L INTRANET
# ++++++++++++++++++++++++++++++++++++++++++++++++
# LES VALVES ELECTRONIQUES
# PAR FR. FILEE
# JUILLET 1999
# ++++++++++++++++++++++++++++++++++++++++++++++++
# ++++++++++++++++++++++++++++++++++++++++++++++++
# SCRIPT AjoutAvisValve(.CGI)
# ++++++++++++++++++++++++++++++++++++++++++++++++
# ++++++++++++++++++++++++++++++++++++++++++++++++

# ++++++++++++++++++++++++++++++++++++++++++++++++
# ++++++++++++++++++++++++++++++++++++++++++++++++
# DECLARATION DES FONCTIONS
# ++++++++++++++++++++++++++++++++++++++++++++++++
# ++++++++++++++++++++++++++++++++++++++++++++++++


# ++++++++++++++++++++++++
function Bisext
# ++++++++++++++++++++++++

# la fonction prend un argument :
#    l'annee dont il faut verifier le caractere bisextile
# elle renvoie 1 si elle est bisextile, 0 sinon

{
declare -i Res1=0

Res1=$1%4

if test $Res1 -eq 0
then
   declare -i Res2=0
   declare -i Res3=0
   Res2=$1%100
   Res3=$1%400
   if test $Res2 -ne 0 -o $Res3 -eq 0
   then
       return 1
   else return 0
   fi
else return 0
fi
}
```

```
# +++++++++++++++++++++++
function CompareDate
# +++++++++++++++++++++++
{

# Cette fonction compare deux dates (Jour1, Mois1, An1 et Jour2, Mois2, An2)
# elle renvoie 1 si Date1 < Date2
# elle renvoie 0 si Date1 > Date2
# elle renvoie 2 si Date1 = Date2

if test $6 -eq $3 -a $5 -eq $2 -a $4 -eq $1
then
   return 2
else
   if test $6 -gt $3
   then
      return 1
   else
      if test $6 -eq $3 -a $5 -gt $2
      then
         return 1
      else
         if test $6 -eq $3 -a $5 -eq $2 -a $4 -gt $1
         then
            return 1
         else
            return 0
         fi
      fi
   fi
fi

}

# +++++++++++++++++++++++
function CreerValve
# +++++++++++++++++++++++

# la fonction prend deux parametres
#   1. le nom generique de la valve (ValveStaff, Valve1LM, Valve2LM, Valve3M)
#   2. le type de valve a creer (Virtuel, Reel)
#   3. le nom du fichier des valves a creer
# son resultat est la creation de la valve correspondante

{
TitreValve=''
case $1 in
  ValveStaff) TitreValve='Valves du staff';;
  Valve1LM) TitreValve='Valves des 1&egrave;res licences et ma&icirc;trises';;
  Valve2LM) TitreValve='Valves des 2&egrave;mes licences et ma&icirc;trises';;
  Valve3M) TitreValve='Valves des 3&egrave;mes ma&icirc;trises';;
esac

echo "<HTML> <HEAD>" > $3
echo "<TITLE>Valves electroniques</TITLE> </HEAD>" >> $3
echo '<BODY BGCOLOR="#fffff0" OnChange="location.reload()"><H1 ALIGN="CENTER">' >> $3

echo $TitreValve >> $3
```

```
if test $2 = 'Virtuel'
then
  echo '</br>Liste des avis en attente de publication' >> $3
fi

echo '</H1>' >> $3
echo '<HR ALIGN="LEFT">' >> $3
echo '' >> $3

echo '<!-- Cree le ' $JourSysteme '/' $MoisSysteme '/' $AnneeSysteme ' -->' >> $3
echo '<!-- par le numero IP suivant' $AuteurAvis ' -->' >> $3
echo '<!--' '======MarqueFinEntete======' '-->' >> $3
echo '<!--' '======MarqueFinValve======' '-->' >> $3
echo "<br><br><HR>" >> $3
echo '<TT><a href="'"$URLIntranet'"><IMG ALIGN=middle SRC="/icons/home.gif"></a> Retour
&agrave;' "l'intranet" >> $3
echo '<a href="'"$URLInstitut'/"><IMG ALIGN=middle SRC="/icons/home.gif"></a> Institut
d'"'Informatique<p> </TT>" >> $3
echo '</BODY></HTML>' >> $3

return 1
}

# +++++++++++++++++++++++
function DecoderDonneesAjoutAvis
# +++++++++++++++++++++++
{

# Les valeurs des variables sont envoyes par HTTP sur le standard input.
# cgiparse decode cet input et genere une suite d'instructions de la
# forme "FORM_variable=valeur_decodee" qui vont etre executees par sh
# et permettre de remplir les variables definies ci-dessus avec les
# bonnes valeurs.

eval `/opt/blanc/comms/httpd/bin/cgiparse -form`
AuteurAvis=$REMOTE_HOST
return 1
}

# +++++++++++++++++++++++
function EmpecherSimultaneite
# +++++++++++++++++++++++
{
ExisteFichier $LockValve
if test $? -eq 1
then
   # le fichier LockValve indique que les valves sont en modification
# le programme principal va provoquer l'arret du script
return 0
else
  echo 'Occupe' > $LockValve
  return 1
fi
}
```

```
# ++++++++++++++++++++++++
function EnvoiMail
# ++++++++++++++++++++++++

# La fonction recoit quatre arguments :
#    1. le nom du destinataire (voir constante)
#    2. le sujet (voir constante)
#    3. Le type d'erreur ('ValveInexistante')
#    4. Le nom du fichier incrimine par 'ValveInexistante'
# Son resultat est l'envoi d'un mail ad hoc

{
case $3 in

   ValveInexistante)

  mail $1 << EOM # ? option mail -t
From: www@info.fundp.ac.be (Script WWW)
Subject: $2

  Le controle automatique des valves a detecte la presence d'une valve INEXISTANTE.
  En theorie, cette situation est impossible.
  Cela revele un probleme de fonctionnement des valves electroniques.

  Le controle automatique a simplement cree le fichier: $4.
  Avec l'espoir que ce probleme est seulement ponctuel.

  === Adresse IP sender : $REMOTE_HOST ===
EOM
      return 1;;

   *) return 0;;
esac
}

# ++++++++++++++++++++++++
function ErreurAjout
# ++++++++++++++++++++++++

# la fonction prend 2 parametres :
#   1. le message a afficher sur le type d'erreur
#   2. un conseil pour l'utilisateur

{
echo "Content-type: text/html"
echo ""
echo "<HTML> <HEAD>"
echo "<TITLE>R&eacute;sultat d'un ajout d'avis aux valves</TITLE> </HEAD>"
echo '<BODY BGCOLOR="#fffff0">'
echo "<h1>Votre demande n'a pas &eacute;t&eacute; accept&eacute;e !</h1><HR>"
echo $1
echo $2
echo "<br><br><HR>"
echo '<TT><a href="'$URLIntranet'"><IMG ALIGN=middle SRC="/icons/home.gif"></a> Retour
&agrave;' "l'intranet"
echo '<a href="'$URLInstitut'/"><IMG ALIGN=middle SRC="/icons/home.gif"></a> Institut
d'"'Informatique<p> </TT>"
echo '</BODY></HTML>'
return 1
}
```

```
# +++++++++++++++++++++++
function ErreurValidation
# +++++++++++++++++++++++

# la fonction prend 1 parametre :
# le message a afficher sur le type d'erreur

{
echo "Content-type: text/html"
echo ""
echo "<HTML> <HEAD>"
echo "<TITLE>R&eacute;sultat d'un ajout d'avis aux valves</TITLE> </HEAD>"
echo '<BODY BGCOLOR="#fffff0">'
echo "<h1>Votre demande n'a pas &eacute;t&eacute; accept&eacute;e !</h1><HR>"
echo $1
echo "<br><br><HR>"
echo '<TT><a href="'$URLIntranet'"><IMG ALIGN=middle SRC="/icons/home.gif"></a> Retour
&agrave;' "l'intranet"
echo '<a href="'$URLInstitut'/"><IMG ALIGN=middle SRC="/icons/home.gif"></a> Institut
d'"'Informatique<p> </TT>"
echo '</BODY></HTML>'

return 1
}

# +++++++++++++++++++++++
function EstVideValve
# +++++++++++++++++++++++
{
declare -i NbreAvis
NbreAvis=`grep -c "====Titre====" $1`

if test $NbreAvis -eq 0
then
   return 1
else
   return 0
fi
}

# +++++++++++++++++++++++
function ExisteFichier
# +++++++++++++++++++++++
{
if test -e $1
then
   return 1
else
   return 0
fi
}
```

```
# +++++++++++++++++++++++
function InsererAvisValve
# +++++++++++++++++++++++
{
# Copie du fichier valve dans fichier intermediaire avec ajout de l avis

declare -i LgueurEntete
LgueurEntete=$CstEntete-1  # par pure precaution

EstVideValve $NomFichier

if test $? -eq 1
then
   Vide=true
else
   Vide=false
fi


Trouve=false

declare -i Cpt=0

while read Ligne
do

  Cpt=Cpt+1

 if test $Cpt -gt $LgueurEntete
 then
  if test $Trouve = false
  then
    declare -i NbMot
    NbMot=`echo $Ligne | wc -w`
    if test $NbMot -lt 2
    then
      Mot='Bidon'
    else
      Mot=`echo $Ligne | cut -f 2-2 -d ' '`
    fi

    case $Mot in

      '======MarqueFinEntete======')

                # Copie de marque de fin d entete
                echo $Ligne >> $NomFichInterm


                # Insertion de l avis comme premier de la liste

                echo '<!-- =====MarqueDebutAvis===== -->' >> $NomFichInterm
                echo ' ' >> $NomFichInterm

                echo '<!-- ====Titre==== --> <H2>'$TitreAvis'</H2></P>' >> $NomFichInterm
                echo '<!-- ====Contenu==== --> <BLOCKQUOTE>'$ContenuAvis'</BLOCKQUOTE></BR>'
>> $NomFichInterm
                echo '<SMALL>Cet avis a &eacute;t&eacute; d&eacute;pos&eacute; pour le' >>
$NomFichInterm
```

```
            echo '<!-- ====DateDepot==== -->' $JourDepotAvis '/' $MoisDepotAvis '/' $AnneeDepotAvis
>> $NomFichInterm
            echo 'et sera retir&eacute; le' >> $NomFichInterm
            echo '<!-- ====DatePeremption==== -->' $JourPeremptionAvis '/' $MoisPeremptionAvis '/'
$AnneePeremptionAvis >> $NomFichInterm
            echo '</SMALL></P>' >> $NomFichInterm
            echo '<!-- ====Auteur==== de cet avis :' $AuteurAvis '-->' >> $NomFichInterm
            echo '<hr ALIGN="LEFT">'>> $NomFichInterm
            echo ' ' >> $NomFichInterm
            echo '<!-- =====MarqueFinAvis===== -->' >> $NomFichInterm

            if test $Vide = false
            then
               Trouve=true
            fi;;

    '=====ValveVide=====')

            # ne pas recopier la ligne : ne rien faire!
            Trouve=true;;

    *)
            # Copie des donnees de la valve precedente
            echo $Ligne >> $NomFichInterm;;
  esac
 else echo $Ligne >> $NomFichInterm
 fi
 else echo $Ligne >> $NomFichInterm
 fi

done < $NomFichier

return 1

}

# +++++++++++++++++++++++
function RestaurationFichier
# +++++++++++++++++++++++
{

# Copie du fichier intermediaire comme nouvelle valve
# et suppression du fichier intermediaire

cp $NomFichInterm $NomFichier
rm $NomFichInterm

return 1

}
```

```
# +++++++++++++++++++++++
function TraiterAvis
# +++++++++++++++++++++++
{

NomFichier=$CheminFichier$Dest$TypeAvis.html

# test de l'existence du fichier de la valve

ExisteFichier $NomFichier
if test $? -eq 0
then
   # le fichier de la valve n'existe pas
   # il faut le creer (afin de garantir la plus grande coherence possible des fichiers)
   # envoi d'un mail au Webmaster
   CreerValve $Dest $TypeAvis $NomFichier
   EnvoiMail $Destinataire $Sujet 'ValveInexistante' $NomFichier
fi

InsererAvisValve

return 1
}

# +++++++++++++++++++++++
function TyperAvis
# +++++++++++++++++++++++
{

CompareDate $JourDepotAvis $MoisDepotAvis $AnneeDepotAvis $JourSysteme $MoisSysteme
$AnneeSysteme
if test $? -eq 0
then
   # la date de depot est strictement posterieure a la date systeme
   TypeAvis='Virtuel'
else
   # la date de depot est anterieure ou egale a la date systeme
   TypeAvis='Reel'
fi

return 1

}
# +++++++++++++++++++++++
function ValideDate
# +++++++++++++++++++++++

# la fonction prend 3 parametres
#    1. le jour
#    2. le mois
#    3. l'annee
# elle renvoie 1 si cela forme une date correcte
#            0 si cela forme une date incorrecte
#         et 2 si la date est anterieure a la date systeme
#                   (la date systeme est un ensemble de trois variables globales
#                    JourSysteme, MoisSysteme, AnneeSysteme)

{

declare -i Jour=$1
```

```
declare -i Mois=$2
declare -i Annee=$3

declare -i Interm=0

if test $Jour -lt 1 -o $Jour -gt 31 -o $Mois -lt 1 -o $Mois -gt 12 -o $Annee -lt 1990 -o $Annee -gt 2050
then
   return 0
else
   if test \( $Mois -eq 4 -o $Mois -eq 6 -o $Mois -eq 9 -o $Mois -eq 11 \) -a \( $Jour -eq 31 \)
   then
      return 0
   else
     if test $Mois -eq 2 -a $Jour -gt 29
     then
         return 0
     else
         Bisext $Annee
         Interm=$?
         if test $Mois -eq 2 -a $Interm -eq 0 -a $Jour -eq 29
         then
            return 0
         else
            CompareDate $Jour $Mois $Annee $JourSysteme $MoisSysteme $AnneeSysteme
            if test $? -eq 1
            then
               return 2
            else
               return 1
            fi
         fi
     fi
   fi
fi

}

# +++++++++++++++++++++++
function ValiderDonneesAvis
# +++++++++++++++++++++++
{
declare -i Result=0
Result=`echo $FORM_TitreAvis | wc -c`
Result=Result-1
if test $Result -eq 0
then

  ErreurValidation "Le titre de l'avis ne peut &ecirc;tre vide"
  return 0
else
  TitreAvis=$FORM_TitreAvis
fi

declare -i Result=0
Result=`echo $FORM_ContenuAvis | wc -c`
Result=Result-1
```

```
if test $Result -eq 0
then
  ErreurValidation "Le contenu de l'avis ne peut &ecirc;tre vide"
  return 0
else
  ContenuAvis=$FORM_ContenuAvis
fi

declare -i Result=0
ValideDate $FORM_DeJour $FORM_DeMois $FORM_DeAnnee
Result=$?
if test $Result -eq 0
then
  ErreurValidation "La date du d&eacute;p&ocirc;t de l'avis n'est pas valide"
  return 0
else
  if test $Result -eq 2
  then
    ErreurValidation "La date du d&eacute;p&ocirc;t de l'avis est ant&eacute;rieure &agrave; celle
d'aujourd'hui"
    return 0
  else
    JourDepotAvis=$FORM_DeJour
    MoisDepotAvis=$FORM_DeMois
    AnneeDepotAvis=$FORM_DeAnnee
  fi
fi

declare -i Result=0
ValideDate $FORM_AuJour $FORM_AuMois $FORM_AuAnnee
Result=$?
if test $Result -eq 0
then
  ErreurValidation "La date du p&eacute;remption de l'avis n'est pas valide"
  return 0
else
  if test $Result -eq 2
  then
    ErreurValidation "La date du p&eacute;remption de l'avis est ant&eacute;rieure &agrave; celle
d'aujourd'hui"
    return 0
  else
    JourPeremptionAvis=$FORM_AuJour
    MoisPeremptionAvis=$FORM_AuMois
    AnneePeremptionAvis=$FORM_AuAnnee
  fi
fi

CompareDate $JourDepotAvis $MoisDepotAvis $AnneeDepotAvis $JourPeremptionAvis
$MoisPeremptionAvis $AnneePeremptionAvis
if test $? -ne 1
then
  # la date de depot est posterieure ou egale a la date de peremption
  ErreurValidation "La date du d&eacute;p&ocirc;t de l'avis doit &ecirc;tre ant&eacute;rieure &agrave;
celle de p&eacute;remption"
  return 0
fi

OkStaff=$FORM_OkStaff
Ok1LM=$FORM_Ok1LM
```

```
Ok2LM=$FORM_Ok2LM
Ok3M=$FORM_Ok3M

DestinataireAvis=''
if test $OkStaff = 'ON'
then
    DestinataireAvis=`echo $DestinataireAvis 'ValveStaff'`
fi
if test $Ok1LM = 'ON'
then
    DestinataireAvis=`echo $DestinataireAvis 'Valve1LM'`
fi
if test $Ok2LM = 'ON'
then
    DestinataireAvis=`echo $DestinataireAvis 'Valve2LM'`
fi
if test $Ok3M = 'ON'
then
    DestinataireAvis=`echo $DestinataireAvis 'Valve3M'`
fi

declare -i Result=0
Result=`echo $DestinataireAvis | wc -c`
Result=Result-1
if test $Result -eq 0
then
   ErreurValidation 'Il faut au moins un  destinataire pour cet avis'
   return 0
else
   return 1
fi
}

# ++++++++++++++++++++++++++++++++++++++++++++++++++
# ++++++++++++++++++++++++++++++++++++++++++++++++++
# INITIALISATION DES VARIABLES GLOBALES
# ++++++++++++++++++++++++++++++++++++++++++++++++++
# ++++++++++++++++++++++++++++++++++++++++++++++++++

# ++++++++++++++++++++++++
# InitialisationConstante
# ++++++++++++++++++++++++

# Envoi de mail pour le webmaster

Destinataire='www@info.fundp.ac.be'
Sujet='ValveElectronique/ControleAutomatique'

# Liens hypertextes de fin de pages Web
URLInstitut='http://www.info.fundp.ac.be'
URLIntranet='http://www.info.fundp.ac.be/~ffilee/intranet.html'

# Chemin de repertoire pour les fichiers valves

CheminFichier='../FichierValve/'

# Fichier Lock des valves
LockValve=$CheminFichier"LockValve"
```

```
# Caracteristiques des valves et avis

# Longueur max de l entete d une valve (nbre de ligne)
declare -i CstEntete
CstEntete=9

# Longueur min d un avis (nbre de ligne)
declare -i CstLigne
CstLigne=12

# +++++++++++++++++++++++
# InitialisationVariables
# +++++++++++++++++++++++

# Initialisation des variables du formulaire


FORM_TitreAvis=''
FORM_ContenuAvis=''
FORM_DeJour=''
FORM_DeMois=''
FORM_DeAnnee=''
FORM_AuJour=''
FORM_AuMois=''
FORM_AuAnnee=''
FORM_OkStaff='OFF'
FORM_Ok1LM='OFF'
FORM_Ok2LM='OFF'
FORM_Ok3M='OFF'

# Initialisation des variables valides issues du formulaire

TitreAvis=''
ContenuAvis=''
declare -i JourDepotAvis
declare -i MoisDepotAvis
declare -i AnneeDepotAvis
declare -i JourPeremptionAvis
declare -i MoisPeremptionAvis
declare -i AnneePeremptionAvis
OkStaff=''
Ok1LM=''
Ok2LM=''
Ok3M=''

DestinataireAvis=''
AuteurAvis=''

# Initialisation des variables fonctionnelles

TypeAvis=''
```

```
# InitialisationDateSysteme

MotJourSysteme=`date +"%d"`
declare -i JourSysteme
case $MotJourSysteme in
        01) JourSysteme=1;;
        02) JourSysteme=2;;
        03) JourSysteme=3;;
        04) JourSysteme=4;;
        05) JourSysteme=5;;
        06) JourSysteme=6;;
        07) JourSysteme=7;;
        08) JourSysteme=8;;
        09) JourSysteme=9;;
        *)  JourSysteme=$MotJourSysteme;;
esac

MotMoisSysteme=`date +"%m"`
declare -i MoisSysteme
case $MotMoisSysteme in
        01) MoisSysteme=1;;
        02) MoisSysteme=2;;
        03) MoisSysteme=3;;
        04) MoisSysteme=4;;
        05) MoisSysteme=5;;
        06) MoisSysteme=6;;
        07) MoisSysteme=7;;
        08) MoisSysteme=8;;
        09) MoisSysteme=9;;
        *)  MoisSysteme=$MotMoisSysteme;;
esac

declare -i AnneeSysteme=`date +"%Y"`

# ++++++++++++++++++++++++
# InitialisationFichier
# ++++++++++++++++++++++++

# Creation d un fichier intermediaire pour recopier les valves

declare -i Cpteur=0
NomFichInterm=$CheminFichier"xxx"
while test -e $NomFichInterm
do
  Cpteur=Cpteur+1
  NomFichInterm=$NomFichInterm$Cpteur
done
```

```
# ++++++++++++++++++++++++++++++++++++++++++++++++
# ++++++++++++++++++++++++++++++++++++++++++++++++
# CORPS DU PROGRAMME
# ++++++++++++++++++++++++++++++++++++++++++++++++
# ++++++++++++++++++++++++++++++++++++++++++++++++

EmpecherSimultaneite
if test $? -eq 0
then
# generation d'un ecran d'avertissement (HTML)
# Arret du script
    ErreurAjout 'Les valves sont en modification actuellement' 'R&eacute;assayez dans quelques
instants'
   exit
fi

DecoderDonneesAjoutAvis

ValiderDonneesAvis
if test $? -eq 0
then
    rm $LockValve
    exit
fi

TyperAvis

for Dest in $DestinataireAvis
do
  TraiterAvis

  RestaurationFichier

done

# ++++++++++++++++++++++++
# LibererAccesAuxValves
# ++++++++++++++++++++++++

rm $LockValve

# ++++++++++++++++++++++++
# CommunicationResultat
# ++++++++++++++++++++++++

echo "Content-type: text/html"
echo ""
echo "<HTML> <HEAD>"
echo "<TITLE>R&eacute;sultat d'un ajout d'avis aux valves</TITLE> </HEAD>"
echo '<BODY BGCOLOR="#fffff0">'
echo "<h1>Votre demande a &eacute;t&eacute; accept&eacute;e !</h1><HR>"
echo "Merci."
echo "<br><br><HR>"
echo '<TT><a href="'$URLIntranet'"><IMG ALIGN=middle SRC="/icons/home.gif"></a> Retour
&agrave;' "l'intranet"
echo '<a href="'$URLInstitut'/"><IMG ALIGN=middle SRC="/icons/home.gif"></a> Institut
d'"'Informatique<p> </TT>"
echo '</BODY></HTML>'

exit
```

Annexe 2.2. Le script ConsultSupprValve

```
#!/opt/blanc/gnu/bin/bash
# ++++++++++++++++++++++++++++++++++++++++++++++++
# APPLICATION DE L INTRANET
# ++++++++++++++++++++++++++++++++++++++++++++++++
# LES VALVES ELECTRONIQUES
# PAR FR. FILEE
# JUILLET 1999
# ++++++++++++++++++++++++++++++++++++++++++++++++
# ++++++++++++++++++++++++++++++++++++++++++++++++
# SCRIPT ConsultSupprValve(.CGI)
# ++++++++++++++++++++++++++++++++++++++++++++++++
# ++++++++++++++++++++++++++++++++++++++++++++++++

# Ce script provoque l'affichage de la valve specifiee
# comme un formulaire HTML, dans lequel on peut choisir
# un avis a supprimer
# ce formulaire declenchera le script SuppressionAvisValve.cgi

# ++++++++++++++++++++++++++++++++++++++++++++++++
# ++++++++++++++++++++++++++++++++++++++++++++++++
# DECLARATION DES FONCTIONS
# ++++++++++++++++++++++++++++++++++++++++++++++++
# ++++++++++++++++++++++++++++++++++++++++++++++++

# ++++++++++++++++++++++++
function CreerValveVide
# ++++++++++++++++++++++++

# la fonction prend trois parametres
#   1. le nom generique de la valve (ValveStaff, Valve1LM, Valve2LM, Valve3M)
#   2. le type de valve a creer (Virtuel, Reel)
#   3. le nom du fichier a creer
# son resultat est la creation de la valve correspondante
# Les variables gloables JourSysteme, MoisSysteme, AnneeSysteme sont requises

{
TitreValve=''
case $1 in
  ValveStaff) TitreValve='Valves du staff';;
  Valve1LM) TitreValve='Valves des 1&egrave;res licences et ma&icirc;trises';;
  Valve2LM) TitreValve='Valves des 2&egrave;mes licences et ma&icirc;trises';;
  Valve3M) TitreValve='Valves des 3&egrave;mes ma&icirc;trises';;
esac

echo "<HTML> <HEAD>" > $3
echo "<TITLE>Valves electroniques</TITLE> </HEAD>" >> $3
echo '<BODY BGCOLOR="#fffff0" OnChange="location.reload()"><H1 ALIGN="CENTER">' >> $3

echo $TitreValve >> $3

if test $2 = 'Virtuel'
then
  echo '</br>Liste des avis en attente de publication' >> $3
fi

echo '</H1>' >> $3
echo '<HR ALIGN="LEFT">' >> $3
echo '' >> $3
```

```
echo '<!-- Cree le ' $JourSysteme '/' $MoisSysteme '/' $AnneeSysteme ' -->' >> $3
echo '<!-- par le numero IP suivant' $AuteurAvis ' -->' >> $3
echo '<!--' '======MarqueFinEntete======' '-->' >> $3
echo '<!--' '=====ValveVide=====' "--> <H2> Aucun avis n'existe sur ces valves actuellement !</H2>"
>> $3
echo '<!--' '======MarqueFinValve======' '-->' >> $3
echo "<br><br><HR>" >> $3
echo '<TT><a href="'"$URLIntranet"'"><IMG ALIGN=middle SRC="/icons/home.gif"></a> Retour
&agrave;' "l'intranet" >> $3
echo '<a href="'"$URLInstitut'/"'><IMG ALIGN=middle SRC="/icons/home.gif"></a> Institut
d'"'Informatique<p> </TT>" >> $3
echo '</BODY></HTML>' >> $3

return 1
}

# +++++++++++++++++++++++
function EmpecherSimultaneite
# +++++++++++++++++++++++
{
ExisteFichier $LockValve
if test $? -eq 1
then
   # le fichier LockValve indique que les valves sont en modification
   # le programme principal va provoquer l'arret du script
   return 0
else
  # ici on ne cree pas le fichier LockValve car la consultation ne modifie pas les valves
   return 1
fi
}

# +++++++++++++++++++++++
function EnregProprietesValve
# +++++++++++++++++++++++

# la fonction recoit deux parametres :
#    1. le nom du fichier des valves concernes
#    2. le nom du fichier des proprietes de ce fichier des valves
# elle provoque l'enregistrement des proprietes du fichier dans le fichier des proprietes

{
echo `ls -l $1` $REMOTE_HOST > $2
return 1
}

# +++++++++++++++++++++++
function EnvoiMail
# +++++++++++++++++++++++

# La fonction recoit quatre arguments :
#    1. le nom du destinataire (voir constante)
#    2. le sujet (voir constante)
#    3. Le type d'erreur ('ValveInexistante')
#    4. Le nom du fichier incrimine par 'ValveInexistante'
# Son resultat est l'envoi d'un mail ad hoc
```

```
{
case $3 in

   ValveInexistante)

  mail $1 << EOM # ? option mail -t
From: www@info.fundp.ac.be (Script WWW)
Subject: $2

  Le controle automatique des valves a detecte la presence d'une valve INEXISTANTE.
  En theorie, cette situation est impossible.
  Cela revele un probleme de fonctionnement des valves electroniques.

  Le controle automatique a simplement cree le fichier: $4.
  Avec l'espoir que ce probleme est seulement ponctuel.

  === Adresse IP sender : $REMOTE_HOST ===
EOM
    return 1;;

   *) return 0;;
esac
}

# +++++++++++++++++++++++
function ErreurSuppression
# +++++++++++++++++++++++

# la fonction prend 2 parametres :
#    1. le message a afficher sur le type d'erreur
#    2. un conseil a donner a l'utilisateur

{
echo "Content-type: text/html"
echo ""
echo "<HTML> <HEAD>"
echo "<TITLE>R&eacute;sultat de la suppression d'un avis</TITLE> </HEAD>"
echo '<BODY BGCOLOR="#fffff0">'
echo "<h1>" $1 "</h1><HR>"
echo $2
echo "<br><br><HR>"
echo '<TT><a href="'$URLIntranet'"><IMG ALIGN=middle SRC="/icons/home.gif"></a> Retour
&agrave;' "l'intranet"
echo '<a href="'$URLInstitut'/"><IMG ALIGN=middle SRC="/icons/home.gif"></a> Institut
d'"'Informatique<p> </TT>"
echo '</BODY></HTML>'

return 1
}
```

```
# +++++++++++++++++++++++
function EstVideValve
# +++++++++++++++++++++++
{
declare -i NbreAvis
NbreAvis=`grep -c "====Titre====" $1`

if test $NbreAvis -eq 0
then
   return 1
else
   return 0
fi
}


# +++++++++++++++++++++++
function ExisteFichier
# +++++++++++++++++++++++
{
if test -e $1
then
   return 1
else
   return 0
fi
}

# +++++++++++++++++++++++
function PublicationHTML
# +++++++++++++++++++++++

# la fonction prend un argument :
#     le nom du fichier a publier via le navigateur

{
declare -i NumAvis
NumAvis=0

declare -i NbreAvis
NbreAvis=`grep -c "====Titre====" $1`

declare -i NumLigneAvis
NumLigneAvis=0

declare -i CpteurLigne
CpteurLigne=0

TrouveTitre=false

declare -i DernierAvis
DernierAvis=0

echo "Content-type: text/html"
echo ""
```

```
while read Ligne
do
   CpteurLigne=CpteurLigne+1

   if test $CpteurLigne -lt $CstEntete
   then
      echo $Ligne

   else

      if test $TrouveTitre = true
      then

         if test $NumAvis -eq $NbreAvis
         then
           DernierAvis=DernierAvis+1

           declare -i temp
           temp=CstLigne-1

           if test $DernierAvis -eq $temp
           then
              echo '<BR>'
              echo '<INPUT TYPE="hidden" NAME="NomValve" VALUE="'$NomValve'">'
              echo '<INPUT TYPE="hidden" NAME="TypeValve" VALUE="'$TypeValve'">'
              echo '<INPUT TYPE="submit" VALUE="Valider" NAME="ValiderBouton"> <INPUT
TYPE="button" VALUE="Annuler" NAME="AnnulerBouton" onClick="history.back()">'
              echo '</BLOCKQUOTE>'
              echo '</FORM>'
              echo '</BODY></HTML>'
              break
           fi
         fi

         NumLigneAvis=NumLigneAvis+1

         if test $NumLigneAvis -gt $CstLigne
         then
           NumAvis=NumAvis+1
           NumLigneAvis=0
           echo '<input type="radio" value="'$NumAvis'" name="NumAvisSuppr">'
           echo $Ligne
         else
           echo $Ligne
         fi

      else
        declare -i NbMot
        NbMot=`echo $Ligne | wc -w`

        if test $NbMot -lt 2
        then
          echo $Ligne
        else
         Mot=`echo $Ligne | cut -f 2-2 -d ' '`
         case $Mot in

           '====Titre====')
                      NumAvis=NumAvis+1
                      echo "</BR><H1>Formulaire de suppression d'un avis </H1>"
```

```
                              echo "<H2>Cochez l'avis &agrave; supprimer et appuyer sur
&quot;Valider&quot;</H2></BR>"
                              echo "<BLOCKQUOTE>"
                              echo '<FORM ACTION="SuppressionAvis.cgi" METHOD="POST"
NAME="SuppressionForm">'
                              # puisque c'est le 1er avis, on check la valeur du bouton radio
                              # (ainsi on est sur qu'au moins un avis est selectionne)
                              echo '<input type="radio" value="'$NumAvis'" checked name="NumAvisSuppr">'
                              echo $Ligne

                              TrouveTitre=true
                              NumLigneAvis=0;;


        *)
                              echo $Ligne;;

      esac

     fi

   fi

  fi

done < $1

return 1
}

# +++++++++++++++++++++++++++++++++++++++++++++++
# +++++++++++++++++++++++++++++++++++++++++++++++
# INITIALISATION DES VARIABLES GLOBALES
# +++++++++++++++++++++++++++++++++++++++++++++++
# +++++++++++++++++++++++++++++++++++++++++++++++

# +++++++++++++++++++++++
# InitialisationConstante
# +++++++++++++++++++++++

# Envoi de mail pour le webmaster

Destinataire='www@info.fundp.ac.be'

Sujet='ValveElectronique/ControleAutomatique'

# Liens hypertextes de fin de pages Web
URLInstitut='http://www.info.fundp.ac.be'
URLIntranet='http://www.info.fundp.ac.be/~ffilee/intranet.html'

# Chemin de repertoire pour les fichiers valves

CheminFichier='../FichierValve/'

# Fichier Lock des valves
LockValve=$CheminFichier"LockValve"

# Longueur max de l entete d une valve (nbre de ligne)
declare -i CstEntete
CstEntete=9
```

```
# Longueur min d un avis (nbre de ligne)
declare -i CstLigne
CstLigne=12

# +++++++++++++++++++++++
# InitialisationVariables
# +++++++++++++++++++++++

# Initialisation variables fonctionnelles

NomValve=$1
TypeValve=$2
NomFichier=$CheminFichier$NomValve$TypeValve.html
NomFichierPropriete=$CheminFichier$NomValve$TypeValve.Prop

# InitialisationDateSysteme

MotJourSysteme=`date +"%d"`
declare -i JourSysteme
case $MotJourSysteme in
	01) JourSysteme=1;;
	02) JourSysteme=2;;
	03) JourSysteme=3;;
	04) JourSysteme=4;;
	05) JourSysteme=5;;
	06) JourSysteme=6;;
	07) JourSysteme=7;;
	08) JourSysteme=8;;
	09) JourSysteme=9;;
	*)  JourSysteme=$MotJourSysteme;;
esac

MotMoisSysteme=`date +"%m"`
declare -i MoisSysteme
case $MotMoisSysteme in
	01) MoisSysteme=1;;
	02) MoisSysteme=2;;
	03) MoisSysteme=3;;
	04) MoisSysteme=4;;
	05) MoisSysteme=5;;
	06) MoisSysteme=6;;
	07) MoisSysteme=7;;
	08) MoisSysteme=8;;
	09) MoisSysteme=9;;
	*)  MoisSysteme=$MotMoisSysteme;;
esac

declare -i AnneeSysteme=`date +"%Y"`
```

```
# ++++++++++++++++++++++++++++++++++++++++++++++++
# ++++++++++++++++++++++++++++++++++++++++++++++++
# CORPS DU PROGRAMME
# ++++++++++++++++++++++++++++++++++++++++++++++++
# ++++++++++++++++++++++++++++++++++++++++++++++++

ExisteFichier $NomFichier
if test $? -eq 0
then
    # La valve demandee pour la suppression d'un avis n'existe pas
    # creation d'une valve VIDE
    # envoi d'un mail au Webmaster
    # generation d'un ecran d'avertissement (HTML) et arret du script

    EnvoiMail $Destinataire $Sujet 'ValveInexistante' $NomFichier
    CreerValveVide $NomValve $TypeValve $NomFichier
    ErreurSuppression "Aucun avis n'existe sur ces valves actuellement !" "A bient&ocirc;t..."
    exit
fi

EmpecherSimultaneite
if test $? -eq 0
then
   # Le fichier LockValve indique que les valves sont en modification
   # generation d' un ecran d'avertissement (HTML)
   # arret du script
   ErreurSuppression 'Les valves sont d&eacute;j&agrave; en modification actuellement'
'Recommencez la proc&eacute;dure dans quelques instants'
   exit

fi

EstVideValve $NomFichier
if test $? -eq 1
then
   # La valve demandee pour la suppression d'un avis ne contient aucun avis
   # generation d'un ecran d'avertissement (HTML)
   # arret du script
   ErreurSuppression "Aucun avis n'existe sur ces valves actuellement !" "A bient&ocirc;t..."
   exit
fi

EnregProprietesValve $NomFichier $NomFichierPropriete

PublicationHTML $NomFichier

exit
```

Annexe 2.3. Le script SuppressionAvis

```
#!/opt/blanc/gnu/bin/bash
# +++++++++++++++++++++++++++++++++++++++++++++++
# APPLICATION DE L INTRANET
# +++++++++++++++++++++++++++++++++++++++++++++++
# LES VALVES ELECTRONIQUES
# PAR FR. FILEE
# JUILLET 1999
# +++++++++++++++++++++++++++++++++++++++++++++++
# +++++++++++++++++++++++++++++++++++++++++++++++
# SCRIPT SuppressionAvis(.CGI)
# +++++++++++++++++++++++++++++++++++++++++++++++
# +++++++++++++++++++++++++++++++++++++++++++++++

# Ce script entraine la suppression de l'avis
# dont le formulaire FormSuppressionAvis (produit par le script ConsultSupprValve.cgi)
# a donne le numero

# +++++++++++++++++++++++++++++++++++++++++++++++
# +++++++++++++++++++++++++++++++++++++++++++++++
# DECLARATION DES FONCTIONS
# +++++++++++++++++++++++++++++++++++++++++++++++
# +++++++++++++++++++++++++++++++++++++++++++++++

# ++++++++++++++++++++++++
function ControleApresVacuiteValve
# ++++++++++++++++++++++++
{
EstVideValve $NomFichier
if test $? -eq 1
then
  # La valve ne plus contient aucun avis
  rm $NomFichier
  CreerValveVide $NomValve $TypeValve $NomFichier
fi

return 1
}

# ++++++++++++++++++++++++
function ControleModifValve
# ++++++++++++++++++++++++

# la fonction prend 2 arguments
#   1. le nom du fichier des valves dont il faut supprimer un avis
#   2. le nom du fichier ou se trouvent les proprietes de ces valves
#        ce fichier de proprietes a ete construit par le script ConsultSupprValve.cgi
#        en effectuant un 'ls' sur le fichier des valves
# cette fonction compare donc les proprietes du fichier des valves lorsqu'il a ete consulte
# sous la forme du formulaire de suppression (SuppressionAvisForm)
# et les proprietes actuelles du fichier des valves
#
# si tout est identique, elle renvoie 1
# si les donnees sont differentes, elle renvoie 0
# si le fichier des proprietes n'existe pas elle renvoie 2
```

```
{
ExisteFichier $2
if test $? -eq 0
then
  return 2
else
  Ls2=`ls -l $1`
  Ls2=`echo $Ls2 $REMOTE_HOST`

  Idem='true'

  while read Ligne
  do
    Ls1=$Ligne
    break
  done < $2

  for Donnee in $Ls1
  # il faut parcourir toute la chaine de caractere mot par mot
  # car on ne peut faire un 'if test' sur des lignes
  do
    if test $Donnee = `echo $Ls2 | cut -f 1-1 -d ' '`
    then
      Ls2=`echo $Ls2 | cut -f 2- -d ' '`
    else
      Idem='false'
      break
    fi
  done


  if test $Idem = 'false'
  then
    # Le fichier des valves a ete modifies
    # depuis que le formulaire de suppression a ete construit et visualise
    return 0
  else
    # Le fichier des valves est toujours le meme depuis sa visualisation
    # dans le formulaire de suppression
    return 1
  fi
fi
}

# +++++++++++++++++++++++
function CreerValveVide
# +++++++++++++++++++++++

# la fonction prend trois parametres
#   1. le nom generique de la valve (ValveStaff, Valve1LM, Valve2LM, Valve3M)
#   2. le type de valve a creer (Virtuel, Reel)
#   3. le nom du fichier a creer
# son resultat est la creation de la valve correspondante
# Les variables gloables JourSysteme, MoisSysteme, AnneeSysteme sont requises
```

```
{
TitreValve=''
case $1 in
  ValveStaff) TitreValve='Valves du staff';;
  Valve1LM) TitreValve='Valves des 1&egrave;res licences et ma&icirc;trises';;
  Valve2LM) TitreValve='Valves des 2&egrave;mes licences et ma&icirc;trises';;
  Valve3M) TitreValve='Valves des 3&egrave;mes ma&icirc;trises';;
esac

echo "<HTML> <HEAD>" > $3
echo "<TITLE>Valves electroniques</TITLE> </HEAD>" >> $3
echo '<BODY BGCOLOR="#fffff0" OnChange="location.reload()"><H1 ALIGN="CENTER">' >> $3

echo $TitreValve >> $3

if test $2 = 'Virtuel'
then
  echo '</br>Liste des avis en attente de publication' >> $3
fi

echo '</H1>' >> $3
echo '<HR ALIGN="LEFT">' >> $3
echo '' >> $3

echo '<!-- Cree le ' $JourSysteme '/' $MoisSysteme '/' $AnneeSysteme ' -->' >> $3
echo '<!-- par le numero IP suivant' $AuteurAvis ' -->' >> $3
echo '<!--' '======MarqueFinEntete======' '-->' >> $3
echo '<!--' '=====ValveVide=====' "--> <H2> Aucun avis n'existe sur ces valves actuellement !</H2>"
>> $3
echo '<!--' '======MarqueFinValve======' '-->' >> $3
echo "<br><br><HR>" >> $3
echo '<TT><a href="'$URLIntranet'"><IMG ALIGN=middle SRC="/icons/home.gif"></a> Retour
&agrave;' "l'intranet" >> $3
echo '<a href="'$URLInstitut'/"><IMG ALIGN=middle SRC="/icons/home.gif"></a> Institut
d'"'Informatique<p> </TT>" >> $3
echo '</BODY></HTML>' >> $3

return 1
}

# ++++++++++++++++++++++++
function DecoderDonneesAvis
# ++++++++++++++++++++++++
{

# Les valeurs des variables sont envoyes par HTTP sur le standard input.
# cgiparse decode cet input et genere une suite d'instructions de la
# forme "FORM_variable=valeur_decodee" qui vont etre executees par sh
# et permettre de remplir les variables definies ci-dessus avec les
# bonnes valeurs.


eval `/opt/blanc/comms/httpd/bin/cgiparse -form`
return 1
}
```

```
# +++++++++++++++++++++++
function EmpecherSimultaneite
# +++++++++++++++++++++++
{
ExisteFichier $LockValve
if test $? -eq 1
then
   # le fichier LockValve indique que les valves sont en modification
   # le programme principal va provoquer l'arret du script
   return 0
else
   echo 'Occupe' > $LockValve
   return 1
fi
}
# +++++++++++++++++++++++
function EnvoiMail
# +++++++++++++++++++++++

# La fonction recoit quatre arguments :
#    1. le nom du destinataire (voir constante)
#    2. le sujet (voir constante)
#    3. Le type d'erreur ('ValveInexistante')
#    4. Le nom du fichier incrimine par 'ValveInexistante'
# Son resultat est l'envoi d'un mail ad hoc

{
case $3 in

  ValveInexistante)

 mail $1 << EOM # ? option mail -t
From: www@info.fundp.ac.be (Script WWW)
Subject: $2

 Le controle automatique des valves a detecte la presence d'une valve INEXISTANTE.
 En theorie, cette situation est impossible.
 Cela revele un probleme de fonctionnement des valves electroniques.

 Le controle automatique a simplement cree le fichier: $4.
 Avec l'espoir que ce probleme est seulement ponctuel.

 === Adresse IP sender : $REMOTE_HOST ===
EOM
   return 1;;

  *) return 0;;
esac
}

# +++++++++++++++++++++++
function ErreurSuppression
# +++++++++++++++++++++++

# la fonction prend 2 parametres :
#    1. le message a afficher sur le type d'erreur
#    2. un conseil a donner a l'utilisateur
```

```
{
echo "Content-type: text/html"
echo ""
echo "<HTML> <HEAD>"
echo "<TITLE>R&eacute;sultat de la suppression d'un avis</TITLE> </HEAD>"
echo '<BODY BGCOLOR="#fffff0">'
echo "<h1>" $1 "</h1><HR>"
echo $2
echo "<br><br><HR>"
echo '<TT><a href="'$URLIntranet'"><IMG ALIGN=middle SRC="/icons/home.gif"></a> Retour
&agrave;' "l'intranet"
echo '<a href="'$URLInstitut'/"><IMG ALIGN=middle SRC="/icons/home.gif"></a> Institut
d'"'Informatique<p> </TT>"
echo '</BODY></HTML>'

return 1
}

# +++++++++++++++++++++++
function EstVideValve
# +++++++++++++++++++++++
{
declare -i NbreAvis
NbreAvis=`grep -c "====Titre====" $1`

if test $NbreAvis -eq 0
then
   return 1
else
   return 0
fi
}

# +++++++++++++++++++++++
function ExisteFichier
# +++++++++++++++++++++++
{
if test -e $1
then
   return 1
else
   return 0
fi
}
# +++++++++++++++++++++++
function SupprimerUnAvis
# +++++++++++++++++++++++

{

# Creation d un fichier intermediaire

declare -i Cpteur=0
NomFichInterm=$CheminFichieryxy
while test -e $NomFichInterm
do
   Cpteur=Cpteur+1
   NomFichInterm=$NomFichInterm$Cpteur
done
```

```
# Copie du fichier valve dans fichier intermediaire sans les avis perimes

declare -i NumLigneDebut
NumLigneDebut=0
declare -i NumLigneFin
NumLigneFin=0

declare -i CpteurLigne
CpteurLigne=0

TrouveAvis=false

while read Ligne
do
   CpteurLigne=CpteurLigne+1

   if test $CpteurLigne -lt $CstEntete
   then
     echo $Ligne >> $NomFichInterm
   else

     if test $TrouveAvis = true
     then
        if test $CpteurLigne -lt $NumLigneDebut -o $CpteurLigne -gt $NumLigneFin
        then
           echo $Ligne >> $NomFichInterm
        fi
     else

       declare -i NbMot
       NbMot=`echo $Ligne | wc -w`
       if test $NbMot -ge 2
       then
         # si la ligne compte au moins deux mots, on prend le deuxième mot
         Mot=`echo $Ligne | cut -f 2-2 -d ' '`
       else
         Mot='Bidon'
       fi

       case $Mot in

           '=====MarqueDebutAvis=====')

                    TrouveAvis=true

                    if test $NumAvisSuppr -eq 1
                    then
                       NumLigneDebut=$CpteurLigne
                       NumLigneFin=$NumLigneDebut+$CstLigne
                    else

                       echo $Ligne >> $NomFichInterm

                       declare -i Temp
                       Temp=$NumAvisSuppr-1
                       NumLigneDebut=$Temp*$CstLigne
                       NumLigneDebut=$NumLigneDebut+$CpteurLigne
                       NumLigneFin=$NumLigneDebut+$CstLigne
                    fi;;
```

```
                    *)

                              echo $Ligne >> $NomFichInterm;;
          esac

        fi
      fi

done < $NomFichier

# RestaurationFichier

cp $NomFichInterm $NomFichier
rm $NomFichInterm

return 1
}

# ++++++++++++++++++++++++
function ValiderDonneesAvis
# ++++++++++++++++++++++++
{

NomValve=$FORM_NomValve

Trouve='false'
for Mot in $ListeValvePossible
# controle si la variable NomValve est bien dans la liste des valves possibles
do
  if test $Mot = $NomValve
  then
    Trouve='true'
    break
  fi
done

if test $Trouve = 'false'
then
   return 0
else
   TypeValve=$FORM_TypeValve

   Trouve='false'
   for Mot in $ListeTypePossible
   # controle si la variable TypeValve est bien dans la liste des types possibles
   do
     if test $Mot = $TypeValve
     then
       Trouve='true'
       break
     fi
   done

   if test $Trouve = 'false'
   then
     return 0
   else
     NumAvisSuppr=$FORM_NumAvisSuppr
```

```
    if test $NumAvisSuppr -lt 0 -o $NumAvisSuppr -gt 99999
    then
      return 0
    else
      return 1
    fi
  fi
fi
}


# ++++++++++++++++++++++++++++++++++++++++++++++++
# ++++++++++++++++++++++++++++++++++++++++++++++++
# INITIALISATION DES VARIABLES GLOBALES
# ++++++++++++++++++++++++++++++++++++++++++++++++
# ++++++++++++++++++++++++++++++++++++++++++++++++

# +++++++++++++++++++++++
# InitialisationConstante
# +++++++++++++++++++++++

ListeValvePossible='ValveStaff Valve1LM Valve2LM Valve3M'
ListeTypePossible='Reel Virtuel'

# Envoi de mail pour le webmaster

Destinataire='www@info.fundp.ac.be'
Sujet='ValveElectronique/ControleAutomatique'

# Liens hypertextes de fin de pages Web

URLInstitut='http://www.info.fundp.ac.be'
URLIntranet='http://www.info.fundp.ac.be/~ffilee/intranet.html'

# Chemin de repertoire pour les fichiers valves

CheminFichier='../FichierValve/'

# Fichier Lock des valves
LockValve=$CheminFichier"LockValve"

# Caracteristiques des valves et avis

# Longueur max de l entete d une valve (nbre de ligne)
declare -i CstEntete
CstEntete=9

# Longueur min d un avis (nbre de ligne)
declare -i CstLigne
CstLigne=12


# +++++++++++++++++++++++
# InitialisationVariables
# +++++++++++++++++++++++

# Initialisation des variables du formulaire

FORM_NomValve=''
FORM_TypeValve=''
```

```
FORM_NumAvisSuppr=''
FORM_ValiderBouton=''

# Initialisation des variables valides issues du formulaire

NomValve=''
TypeValve=''
declare -i NumAvisSuppr=0

# Initialisation des variables fonctionnelles

NomFichier=''
NomFichierPropriete=''

declare -i Result=0

# InitialisationDateSysteme

MotJourSysteme=`date +"%d"`
declare -i JourSysteme
case $MotJourSysteme in
	01) JourSysteme=1;;
	02) JourSysteme=2;;
	03) JourSysteme=3;;
	04) JourSysteme=4;;
	05) JourSysteme=5;;
	06) JourSysteme=6;;
	07) JourSysteme=7;;
	08) JourSysteme=8;;
	09) JourSysteme=9;;
	 *)  JourSysteme=$MotJourSysteme;;
esac

MotMoisSysteme=`date +"%m"`
declare -i MoisSysteme
case $MotMoisSysteme in
	01) MoisSysteme=1;;
	02) MoisSysteme=2;;
	03) MoisSysteme=3;;
	04) MoisSysteme=4;;
	05) MoisSysteme=5;;
	06) MoisSysteme=6;;
	07) MoisSysteme=7;;
	08) MoisSysteme=8;;
	09) MoisSysteme=9;;
	 *)  MoisSysteme=$MotMoisSysteme;;
esac

declare -i AnneeSysteme=`date +"%Y"`
```

```
# ++++++++++++++++++++++++++++++++++++++++++++++++
# ++++++++++++++++++++++++++++++++++++++++++++++++
# CORPS DU PROGRAMME
# ++++++++++++++++++++++++++++++++++++++++++++++++
# ++++++++++++++++++++++++++++++++++++++++++++++++

DecoderDonneesAvis

ValiderDonneesAvis
if test $? -eq 0
then
    # les donnees ne sont pas valides
    # generation d'un ecran d'avertissement (HTML) et arret du script
    ErreurSuppression "Des donn&eacute;es totalement incorrectes ont &eacute;t&eacute; transmises
au serveur" "Recommencez la proc&eacute;dure en rechargeant le formulaire de suppression d'un
avis"
    exit
fi

NomFichier=$CheminFichier$NomValve$TypeValve.html
NomFichierPropriete=$CheminFichier$NomValve$TypeValve.Prop

ExisteFichier $NomFichier
if test $? -eq 0
then
    # La valve demandee pour la suppression d'un avis n'existe plus
    # creation d'une valve VIDE
    # envoi d'un mail au Webmaster
    # generation d'un ecran d'avertissement (HTML) et arret du script
    EnvoiMail $Destinataire $Sujet 'ValveInexistante' $NomFichier
    CreerValveVide $NomValve $TypeValve $NomFichier
    ErreurSuppression "Votre demande n'a pas pu &ecirc;tre ex&eacute;t&eacute;es !" "En effet, ces
valves sont d&eacute;sormais vides !"
    exit
fi

EmpecherSimultaneite
if test $? -eq 0
then
   # Le fichier LockValve indique que les valves sont en modification
   # generation d' un ecran d'avertissement (HTML)
   # arret du script
   ErreurSuppression 'Les valves sont d&eacute;j&agrave; en modification actuellement'
'Recommencez la proc&eacute;dure dans quelques instants'
   exit
fi

ControleModifValve $NomFichier $NomFichierPropriete
Result=$?
if test $Result -eq 0 -o $Result -eq 2
then
   # La valve demandee pour la suppression d'un avis
   # a ete modifiee depuis sa visualisation dans le formulaire de suppression
   # generation d'un ecran d'avertissement (HTML)
   # effacement de LockValve
   # arret du script
   rm $LockValve
```

```
   ErreurSuppression "Votre demande n'a pas pu &ecirc;tre accept&eacute;e ! Car les valves viennent
de subir une modification." "Recommencez la proc&eacute;dure en rechargeant le formulaire
actualis&eacute; de suppression d'un avis"
   exit
fi

SupprimerUnAvis

ControleApresVacuiteValve

# ++++++++++++++++++++++++
# LibererAccesValves
# ++++++++++++++++++++++++

rm $LockValve
rm $NomFichierPropriete

# ++++++++++++++++++++++++
# CommunicationResultat
# ++++++++++++++++++++++++

echo "Content-type: text/html"
echo ""
echo "<HTML> <HEAD>"
echo "<TITLE>R&eacute;sultat de la suppression d'un avis</TITLE> </HEAD>"
echo '<BODY BGCOLOR="#fffff0">'
echo "<h1> Votre demande a &eacute;t&eacute; accept&eacute;e !</h1><HR>"
echo "Merci !"
echo "<br><br><HR>"
echo '<TT><a href="'$URLIntranet'"><IMG ALIGN=middle SRC="/icons/home.gif"></a> Retour
&agrave;' "l'intranet"
echo '<a href="'$URLInstitut'/"><IMG ALIGN=middle SRC="/icons/home.gif"></a> Institut
d"'Informatique<p> </TT>"
echo '</BODY></HTML>'

exit
```

Annexe 2.4. Le script MiseAJourValve

```
#!/opt/blanc/gnu/bin/bash
# +++++++++++++++++++++++++++++++++++++++++++++++++++
# APPLICATION DE L INTRANET
# +++++++++++++++++++++++++++++++++++++++++++++++++++
# LES VALVES ELECTRONIQUES
# PAR FR. FILEE
# JUILLET 1999
# +++++++++++++++++++++++++++++++++++++++++++++++++++
# +++++++++++++++++++++++++++++++++++++++++++++++++++
# SCRIPT MiseAJourValve(.CGI)
# +++++++++++++++++++++++++++++++++++++++++++++++++++
# +++++++++++++++++++++++++++++++++++++++++++++++++++

# +++++++++++++++++++++++++++++++++++++++++++++++++++
# +++++++++++++++++++++++++++++++++++++++++++++++++++
# DECLARATION DES FONCTIONS
# +++++++++++++++++++++++++++++++++++++++++++++++++++
# +++++++++++++++++++++++++++++++++++++++++++++++++++

# +++++++++++++++++++++++++
function CompareDate
# +++++++++++++++++++++++++
{

# Cette fonction compare deux dates (Jour1, Mois1, An1 et Jour2, Mois2, An2)
# elle renvoie 1 si Date1 < Date2
# elle renvoie 0 si Date1 > Date2
# elle renvoie 2 si Date1 = Date2

if test $6 -eq $3 -a $5 -eq $2 -a $4 -eq $1
then
   return 2
else
   if test $6 -gt $3
   then
      return 1
   else
      if test $6 -eq $3 -a $5 -gt $2
      then
         return 1
      else
         if test $6 -eq $3 -a $5 -eq $2 -a $4 -gt $1
         then
            return 1
         else
            return 0
         fi
      fi
   fi
fi

}
```

```
# +++++++++++++++++++++++
function ControleApresVacuiteValve
# +++++++++++++++++++++++
{

# ControleVideValveReelRestante

EstVideValve $NomFichierReel
if test $? -eq 1
then
   # La valve ne plus contient aucun avis
   # il faut creer un fichier de valve vide
   CreerValveVide $NomValve 'Reel' $NomFichierReel
fi


# ControleVideValveVirtuelRestante

if test $Traitement != 'MAJPerime'
then
    EstVideValve $NomFichierVirtuel
    if test $? -eq 1
    then
   # La valve ne plus contient aucun avis
   # il faut creer un fichier de valve vide
   CreerValveVide $NomValve 'Virtuel' $NomFichierVirtuel

    fi
fi

return 1

}

# +++++++++++++++++++++++
function CreerValve
# +++++++++++++++++++++++

# la fonction prend deux parametres
#   1. le nom generique de la valve (ValveStaff, Valve1LM, Valve2LM, Valve3M)
#   2. le type de valve a creer (Virtuel, Reel)
#   3. le nom du fichier des valves a creer
# son resultat est la creation de la valve correspondante

{
TitreValve=''
case $1 in
   ValveStaff) TitreValve='Valves du staff';;
   Valve1LM) TitreValve='Valves des 1&egrave;res licences et ma&icirc;trises';;
   Valve2LM) TitreValve='Valves des 2&egrave;mes licences et ma&icirc;trises';;
   Valve3M) TitreValve='Valves des 3&egrave;mes ma&icirc;trises';;
esac

echo "<HTML> <HEAD>" > $3
echo "<TITLE>Valves electroniques</TITLE> </HEAD>" >> $3
echo '<BODY BGCOLOR="#fffff0" OnChange="location.reload()"><H1 ALIGN="CENTER">' >> $3

echo $TitreValve >> $3
```

```
if test $2 = 'Virtuel'
then
   echo '</br>Liste des avis en attente de publication' >> $3
fi

echo '</H1>' >> $3
echo '<HR ALIGN="LEFT">' >> $3
echo '' >> $3

echo '<!-- Cree le ' $JourSysteme '/' $MoisSysteme '/' $AnneeSysteme ' -->' >> $3
echo '<!-- par le numero IP suivant' $AuteurAvis ' -->' >> $3
echo '<!--' '======MarqueFinEntete======' '-->' >> $3
echo '<!--' '======MarqueFinValve======' '-->' >> $3
echo "<br><br><HR>" >> $3
echo '<TT><a href="'"$URLIntranet"'"><IMG ALIGN=middle SRC="/icons/home.gif"></a> Retour
&agrave;' "l'intranet" >> $3
echo '<a href="'"$URLInstitut'/"><IMG ALIGN=middle SRC="/icons/home.gif"></a> Institut
d"'"Informatique<p> </TT>" >> $3
echo '</BODY></HTML>' >> $3


return 1
}

# ++++++++++++++++++++++++
function CreerValveVide
# ++++++++++++++++++++++++

# la fonction prend trois parametres
#   1. le nom generique de la valve (ValveStaff, Valve1LM, Valve2LM, Valve3M)
#   2. le type de valve a creer (Virtuel, Reel)
#   3. le nom du fichier a creer
# son resultat est la creation de la valve correspondante
# Les variables gloables JourSysteme, MoisSysteme, AnneeSysteme sont requises

{
TitreValve=''
case $1 in
   ValveStaff) TitreValve='Valves du staff';;
   Valve1LM) TitreValve='Valves des 1&egrave;res licences et ma&icirc;trises';;
   Valve2LM) TitreValve='Valves des 2&egrave;mes licences et ma&icirc;trises';;
   Valve3M) TitreValve='Valves des 3&egrave;mes ma&icirc;trises';;
esac

echo "<HTML> <HEAD>" > $3
echo "<TITLE>Valves electroniques</TITLE> </HEAD>" >> $3
echo '<BODY BGCOLOR="#fffff0" OnChange="location.reload()"><H1 ALIGN="CENTER">' >> $3

echo $TitreValve >> $3

if test $2 = 'Virtuel'
then
   echo '</br>Liste des avis en attente de publication' >> $3
fi

echo '</H1>' >> $3
echo '<HR ALIGN="LEFT">' >> $3
echo '' >> $3

echo '<!-- Cree le ' $JourSysteme '/' $MoisSysteme '/' $AnneeSysteme ' -->' >> $3
```

```
echo '<!-- par le numero IP suivant' $AuteurAvis ' -->' >> $3
echo '<!--' '======MarqueFinEntete======' '-->' >> $3
echo '<!--' '=====ValveVide=====' "--> <H2> Aucun avis n'existe sur ces valves actuellement !</H2>"
>> $3
echo '<!--' '======MarqueFinValve======' '-->' >> $3
echo "<br><br><HR>" >> $3
echo '<TT><a href="'$URLIntranet'"><IMG ALIGN=middle SRC="/icons/home.gif"></a> Retour
&agrave;' "l'intranet" >> $3
echo '<a href="'$URLInstitut'/"><IMG ALIGN=middle SRC="/icons/home.gif"></a> Institut
d'"'Informatique<p> </TT>" >> $3
echo '</BODY></HTML>' >> $3

return 1
}

# +++++++++++++++++++++++
function DeplacerAvisActivable
# +++++++++++++++++++++++
{

# Copie des avis activables vers valves tampons
# et copie du fichier valve dans fichier intermediaire sans les avis activables

declare -i CpteurNbreAvis=0
TestActivable='false'

while read Ligne
do
   declare -i NbMot
   NbMot=`echo $Ligne | wc -w`
   if test $NbMot -ge 2
   then
     # si la ligne compte au moins deux mots, on prend le deuxieme mot
     Mot=`echo $Ligne | cut -f 2-2 -d ' '`
   else
     Mot='Bidon'
   fi

   case $Mot in

       '=====MarqueDebutAvis=====')

                  CpteurNbreAvis=CpteurNbreAvis+1
                  declare -i Num
                  TestActivable='false'
                  for Num in $ListeAvisActivable
                  do
                    if test $CpteurNbreAvis -eq $Num
                    then
                      TestActivable='true'
                      break
                    fi
                  done

                  if test $TestActivable = 'true'
                  then
                    echo $Ligne >> $NomValveInterm
                  else
                    echo $Ligne >> $NomFIntermVirtuel
                  fi;;
```

```
        '======MarqueFinValve======')

                    TestActivable='false' # ne servira plus a rien, mais c'est plus logique
                    echo $Ligne >> $NomFIntermVirtuel;;

        *)
                    if test $TestActivable = 'true'
                    then
                       echo $Ligne >> $NomValveInterm
                    else
                      echo $Ligne >> $NomFIntermVirtuel
                    fi;;

    esac

done < $NomFichierVirtuel
}

# +++++++++++++++++++++++
function EmpecherSimultaneite
# +++++++++++++++++++++++
{
if test -e $LockValve
then

   # controle de LockValve

   Ls1LockValve=`ls -l $LockValve`
   sleep 3m
   if test -e $LockValve
   then
      Ls2LockValve=`ls -l $LockValve`
      Idem='true'
      for Donnee in $Ls1LockValve
      do
        if test $Donnee = `echo $Ls2LockValve | cut -f 1-1 -d ' '`
        then
             Ls2LockValve=`echo $Ls2LockValve | cut -f 2- -d ' '`
       else
            Idem='false'
            break
        fi
      done

      if test $Idem = 'false'
      then
         # Le fichier LockValve a ete modifie durant les 3 dernieres minutes
          # on relancera le script 10 minutes plus tard
          at now + 10 minutes << EOF
`echo $NomScript`
EOF
           return 0
```

```
       else
          # Le fichier LockValve est reste identique pendant 3 minutes
          # on envoie un mail au WebMaster et on continue le script

          EnvoiMail $Destinataire $Sujet PresenceLockValve
          return 1

      fi

   else
      echo 'Occupe' > $LockValve
      return 1
   fi
else
    echo 'Occupe' > $LockValve
    return 1
fi
}

# +++++++++++++++++++++++
function EnvoiMail
# +++++++++++++++++++++++

# La fonction recoit quatre arguments :
#    1. le nom du destinataire (voir constante)
#    2. le sujet (voir constante)
#    3. Le type d'erreur ('ValveInexistante', 'PresenceLockValve')
#    4. Eventuellement le nom du fichier incrimine (pour 'ValveInexistante')
# Son resultat est l'envoi d'un mail ad hoc

{

case $3 in

   ValveInexistante)

  mail $1 << EOM # ? option mail -t
From: www@info.fundp.ac.be (Script WWW)
Subject: $2

  Le controle automatique des valves a detecte la presence d'une valve INEXISTANTE.
  En theorie, cette situation est impossible.
  Cela revele un probleme de fonctionnement des valves electroniques.

  Le controle automatique a simplement cree le fichier: $4.
  Avec l'espoir que ce probleme est seulement ponctuel.

  === Adresse IP sender : $REMOTE_HOST ===
EOM
    return 1;;

   PresenceLockValve)

   mail $Destinataire << EOM                        # ? option mail -t
From: www@info.fundp.ac.be (Script WWW)
Subject: $Sujet

   Le controle automatique des valves a detecte la presence anormale
   du fichier $LockValve sur le serveur.
```

```
   La presence de ce fichier empeche normalement le travail simultane sur les valves.
   La presence de ce fichier indique qu'un script
     d'ajout d'un avis aux valves,
     de suppression d'un avis aux valves
     ou de mise a jour des valves,
   n'a pas pu terminer son travail correctement.

   Les valves electroniques risquent donc de ne plus etre coherentes.

   === Adresse IP sender : $REMOTE_HOST ===
EOM
return 1;;

   *) return 0;;
esac
}

# ++++++++++++++++++++++++
function EstVideValve
# ++++++++++++++++++++++++
{
declare -i NbreAvis
NbreAvis=`grep -c "====Titre====" $1`

if test $NbreAvis -eq 0
then
    return 1
else
    return 0
fi
}

# ++++++++++++++++++++++++
function EtablirListeValve
# ++++++++++++++++++++++++

# la fonction cherche des valves existantes
#    si 1 valve virtuelle ou 1 valve reelle est trouvee,
#    elle ajoute le nom generique de la valve a la variable ListeValveExistant
#    pour chaque valves est inexsitante,
#    la fonction créée une valve Vide
# elle renvoie 1 si la liste des valves existantes n'est pas vide
#              0 si la liste des valves existantes est vide

{
declare -i ResReel=0
declare -i ResVirtuel=0
declare -i Res=0

for Mot in $ListeValvePossible
do
  NomReel=$CheminFichier$Mot'Reel.html'
  NomVirtuel=$CheminFichier$Mot'Virtuel.html'

  ExisteFichier $NomReel
  ResReel=$?
  ExisteFichier $NomVirtuel
  ResVirtuel=$?
```

```
  if test $ResReel -eq 1 -o $ResVirtuel -eq 1
  then
    ListeValveExistant=`echo $ListeValveExistant $Mot`
  fi

  if test $ResReel -eq 0
  then
    # le fichier des valves est inexistant
    # creation du fichier des valves
    # envoi d'un mail au webmaster
    CreerValveVide $Mot 'Reel' $NomReel
    EnvoiMail $Destinataire $Sujet 'ValveInexistante' $NomReel
  fi

  if test $ResVirtuel -eq 0
  then
    # le fichier des valves est inexistant
    # creation du fichier des valves
    # envoi d'un mail au webmaster
    CreerValveVide $Mot 'Virtuel' $NomVirtuel
    EnvoiMail $Destinataire $Sujet 'ValveInexistante' $NomVirtuel
 fi

done

Res=`echo $ListeValveExistant | wc -c`
Res=Res-1
if test $Res -eq 0
then
  return 0
else
  return 1
fi
}

# +++++++++++++++++++++++
function ExisteFichier
# +++++++++++++++++++++++
{
if test -e $1
then
   return 1
else
   return 0
fi
}

# +++++++++++++++++++++++
function IdentifierTraitement
# +++++++++++++++++++++++
{
declare -i ResReel=0
declare -i ResVirtuel=0

EstVideValve $NomFichierReel
ResReel=$?
EstVideValve $NomFichierVirtuel
ResVirtuel=$?
```

```
if test $ResReel -eq 0 -a $ResVirtuel -eq 0
then
  Traitement='MAJTotal'
  return 1
else
  if test $ResReel -eq 0 -a $ResVirtuel -eq 1
  then
    Traitement='MAJPerime'
    return 1
  else
    if test $ResReel -eq 1 -a $ResVirtuel -eq 0
    then
       Traitement='MAJActualise'
       return 1
    else
       # aucun traitement sur ces valves
       return 0
    fi
  fi
fi

}

# +++++++++++++++++++++++
function ListerAvisActivable
# +++++++++++++++++++++++
{

declare -i NumAvis=0

ListeAvisActivable=''

while read Ligne
do
 declare -i NbMot
 NbMot=`echo $Ligne | wc -w`
 if test $NbMot -ge 2
 then
   Mot=`echo $Ligne | cut -f 2-2 -d ' '`
   if test $Mot = '====DateDepot===='
   then
       NumAvis=NumAvis+1

       declare -i JourDepot
       JourDepot=`echo $Ligne | cut -f 4-4 -d ' '`
       declare -i MoisDepot
       MoisDepot=`echo $Ligne | cut -f 6-6 -d ' '`
       declare -i AnneeDepot
       AnneeDepot=`echo $Ligne | cut -f 8-8 -d ' '`

       declare -i AvisActivable
       CompareDate $JourDepot $MoisDepot $AnneeDepot $JourSysteme $MoisSysteme
$AnneeSysteme
       AvisActivable=$?
```

```
        if test $AvisActivable -eq 1 -o $AvisActivable -eq 2
        then
           ListeAvisActivable=`echo $ListeAvisActivable $NumAvis`
        fi

    fi
 fi

done < $NomFichierVirtuel
return 1
}

# +++++++++++++++++++++++
function ListerAvisPerime
# +++++++++++++++++++++++
{
declare -i NumAvis=0

ListeAvisPerime=''

while read Ligne
do
  declare -i NbMot
  NbMot=`echo $Ligne | wc -w`
  if test $NbMot -ge 2
  then
    Mot=`echo $Ligne | cut -f 2-2 -d ' '`
    if test $Mot = '====DatePeremption===='
    then
        NumAvis=NumAvis+1

        declare -i JourPeremption
        JourPeremption=`echo $Ligne | cut -f 4-4 -d ' '`
        declare -i MoisPeremption
        MoisPeremption=`echo $Ligne | cut -f 6-6 -d ' '`
        declare -i AnneePeremption
        AnneePeremption=`echo $Ligne | cut -f 8-8 -d ' '`

        declare -i AvisPerime
        CompareDate $JourPeremption $MoisPeremption $AnneePeremption $JourSysteme
$MoisSysteme $AnneeSysteme
       AvisPerime=$?

        if test $AvisPerime -eq 1
        then
           ListeAvisPerime=`echo $ListeAvisPerime $NumAvis`
        fi
    fi
 fi

done < $NomFichierReel

return 1
}
```

```
# +++++++++++++++++++++++++
function MiseAJourValveReelle
# +++++++++++++++++++++++++
{
if test $Traitement = 'MAJActualise'
then
  CreerValve $NomValve 'Reel' $NomFichierReel
fi

# Copie du fichier valve dans fichier intermediaire sans les avis perimes
# avec ajout des avis du fichier valveTampon comme premiers avis

declare -i CpteurNbreAvis=0
TestPerime='false'

while read Ligne
do
   declare -i NbMot
   NbMot=`echo $Ligne | wc -w`
   if test $NbMot -ge 2
   then
     # si la ligne compte au moins deux mots, on prend le deuxieme mot
     Mot=`echo $Ligne | cut -f 2-2 -d ' '`
   else
     Mot='Bidon'
   fi

   case $Mot in

      '======MarqueFinEntete======')


                if test $Traitement != 'MAJPerime'       # Le traitement est MAJActualise ou MAJTotal
                then
                    # Copie de marque de fin d entete
                    echo $Ligne >> $NomFichReelInterm

                    if test -e $NomValveInterm          # car il se peut qu'aucun avis virtuel ne soit
activable
                    then
                       # Insertion des nouveaux avis comme premiers de la liste
                       while read LigneAvis
                        do
                          echo $LigneAvis >> $NomFichReelInterm
                        done < $NomValveInterm
                     fi
                else
                   echo $Ligne >> $NomFichReelInterm
                fi;;

      '=====MarqueDebutAvis=====')

               if test $Traitement != 'MAJActualise'      # Le traitement est MAJPerime ou MAJTotal
               then
                 CpteurNbreAvis=CpteurNbreAvis+1
                 declare -i Num
                 TestPerime='false'
                 for Num in $ListeAvisPerime
                 do
                   if test $CpteurNbreAvis -eq $Num
```

```
              then
                TestPerime='true'
                break
              fi
            done

            if test $TestPerime != 'true'
            then
              echo $Ligne >> $NomFichReelInterm
            fi
          fi;;

    '======MarqueFinValve======')

          if test $Traitement != 'MAJActualise'    # Le traitement est MAJPerime ou MAJTotal
          then
            TestPerime='false' # ne servira plus a rien, mais c'est plus logique
            echo $Ligne >> $NomFichReelInterm
          else
            echo $Ligne >> $NomFichReelInterm
          fi;;

    *)
          if test $Traitement != 'MAJActualise'    # Le traitement est MAJPerime ou MAJTotal
          then
            if test $TestPerime != 'true'
            then
              echo $Ligne >> $NomFichReelInterm
            fi
          else
            echo $Ligne >> $NomFichReelInterm
          fi;;

  esac

done < $NomFichierReel
return 1
}

# +++++++++++++++++++++++
function RestaurationFichiers
# +++++++++++++++++++++++
{
cp $NomFichReelInterm $NomFichierReel
rm $NomFichReelInterm

if test $Traitement != 'MAJPerime'
then
  cp $NomFIntermVirtuel $NomFichierVirtuel
  rm $NomFIntermVirtuel

  if test -e $NomValveInterm # si aucun avis virtuel n'etait activable ce fichier n'existe pas
  then
    rm $NomValveInterm
  fi
fi
return 1
}
```

```
# +++++++++++++++++++++++
function TraiterValve
# +++++++++++++++++++++++
{

  # InitialisationVariables

  NomFichierVirtuel=$CheminFichier$NomValve"Virtuel.html"
  NomFichierReel=$CheminFichier$NomValve"Reel.html"
  Traitement=''

  declare -i Result

  # Identification du traitement a appliquer a la valve

  IdentifierTraitement

  if test $? -eq 0
  then
    return 0
  else

     # InitialisationFichiers

     # Creation d un fichier intermediaire pour recopier les valves reelles
     declare -i Cpteur=0
     NomFichReelInterm=$CheminFichier"xyz"
     while test -e $NomFichReelInterm
     do
        Cpteur=Cpteur+1
        NomFichReelInterm=$NomFichReelInterm$Cpteur
     done

     if test $Traitement != 'MAJPerime'
     then

       # Creation d un fichier intermediaire pour les avis deplaces

       declare -i Cpteur=0
       NomValveInterm=$CheminFichier$NomValve'Tampon'
       while test -e $NomValveInterm
       do
         Cpteur=Cpteur+1
         NomValveInterm=$NomValveInterm$Cpteur
       done

       # Creation d un fichier intermediaire pour recopier les valves virtuelles

       declare -i Cpteur=0
       NomFIntermVirtuel=$CheminFichier"zzz"
       while test -e $NomFIntermVirtuel
       do
          Cpteur=Cpteur+1
          NomFIntermVirtuel=$NomFIntermVirtuel$Cpteur
       done

     fi
```

```
      # listing des avis selon traitement et 1er traitement du virtuel

      if test $Traitement != 'MAJActualise'  # le traitement est MAJPerime ou MAJTotal
      then
         ListerAvisPerime
      fi

      if test $Traitement != 'MAJPerime'  # le traitement est MAJActualise ou MAJTotal
      then
         ListerAvisActivable
         DeplacerAvisActivable
      fi

      # Mise a jour des valves

      MiseAJourValveReelle

      # Remise en ordre de la situation
      RestaurationFichiers

      # Controle a posteriori de fichier vide
      ControleApresVacuiteValve

      # retour normal de la fonction
      return 1
   fi
}

# ++++++++++++++++++++++++++++++++++++++++++++++++
# ++++++++++++++++++++++++++++++++++++++++++++++++
# INITIALISATION DES VARIABLES GLOBALES
# ++++++++++++++++++++++++++++++++++++++++++++++++
# ++++++++++++++++++++++++++++++++++++++++++++++++

# ++++++++++++++++++++++++
# InitialisationConstante
# ++++++++++++++++++++++++

ListeValvePossible='ValveStaff Valve1LM Valve2LM Valve3M'

# Envoi de mail pour le webmaster

Destinataire='www@info.fundp.ac.be'
Sujet='ValveElectronique/ControleAutomatique'

# Liens hypertextes de fin de pages Web
URLInstitut='http://www.info.fundp.ac.be'
URLIntranet='http://www.info.fundp.ac.be/~ffilee/intranet.html'

# Chemin de repertoire pour les fichiers valves

CheminFichier='../FichierValve/'

# Fichier Lock des valves
LockValve=$CheminFichier"LockValve"
```

```
# +++++++++++++++++++++++
# InitialisationVariables
# +++++++++++++++++++++++

NomScript=$0

ListeAvisActivable=''
ListeAvisPerime=''
ListeValveExistant=''

# +++++++++++++++++++++++
# InitialisationDateSysteme
# +++++++++++++++++++++++

MotJourSysteme=`date +"%d"`
declare -i JourSysteme
case $MotJourSysteme in
	01) JourSysteme=1;;
	02) JourSysteme=2;;
	03) JourSysteme=3;;
	04) JourSysteme=4;;
	05) JourSysteme=5;;
	06) JourSysteme=6;;
	07) JourSysteme=7;;
	08) JourSysteme=8;;
	09) JourSysteme=9;;
	 *)  JourSysteme=$MotJourSysteme;;
esac

MotMoisSysteme=`date +"%m"`
declare -i MoisSysteme
case $MotMoisSysteme in
	01) MoisSysteme=1;;
	02) MoisSysteme=2;;
	03) MoisSysteme=3;;
	04) MoisSysteme=4;;
	05) MoisSysteme=5;;
	06) MoisSysteme=6;;
	07) MoisSysteme=7;;
	08) MoisSysteme=8;;
	09) MoisSysteme=9;;
	*)  MoisSysteme=$MotMoisSysteme;;
esac

declare -i AnneeSysteme=`date +"%Y"`
```

```
# +++++++++++++++++++++++++++++++++++++++++++++++++++
# +++++++++++++++++++++++++++++++++++++++++++++++++++
# CORPS DU PROGRAMME
# +++++++++++++++++++++++++++++++++++++++++++++++++++
# +++++++++++++++++++++++++++++++++++++++++++++++++++

EtablirListeValve
if test $? -eq 0
then
  # La liste des valves est vide : arret du script
   exit
fi

EmpecherSimultaneite
if test $? -eq 0
then
  # Les valves sont en modification, le script reprendra dix minutes plus tard
   exit
fi

for NomValve in $ListeValveExistant
do
  TraiterValve
done

# +++++++++++++++++++++++++
# LibererAccesAuxValves
# +++++++++++++++++++++++++

rm $LockValve
```

page blanche

Annexe 3. Code du formulaire d'ajout d'un avis

```
<html>

<head>
<title>Ajout d'un nouvel avis aux valves</title>
<script language="JavaScript">

<!-- CACHE SCRIPT AUX ANCIENS NAVIGATEURS

function TDate(Jour, Mois, Annee)
 {

   this.Jour = parseInt(Jour);
   this.Mois = parseInt(Mois);
   this.Annee = (parseInt(Annee) < 100) ? parseInt(Annee)+1900 : parseInt(Annee);
   this.Bisext = Bisext;
   this.ValideDate = ValideDate;
 }

var today = new Date()

var Maintenant = new TDate(today.getDate(), today.getMonth()+1, (today.getYear() >=2000) ?
today.getYear() : today.getYear() +1900)

var NbJour = 0

function Bisext()
  {
    if ((this.Annee%4==0) && ((this.Annee%100!=0) || (this.Annee%400==0)))
       {
        return true;
       }
       else
       {
       return false;
       }
  }

function CompareDate(Date1, Date2)
  // la fonction renvoie true si la date n°1 est antérieure à la n°2
  // sinon, ou si ces deux dates sont égales, la fonction renvoie false
  {
   DateObjet1 = new Date(Date1.Annee, Date1.Mois-1, Date1.Jour)
   DateObjet2 = new Date(Date2.Annee, Date2.Mois-1, Date2.Jour)

   NbMsJour = 1000 * 60 * 60 * 24

   NbJour = Math.round((DateObjet1.getTime() - DateObjet2.getTime())/NbMsJour);

   if (NbJour >= 0)
     {
      return false;
     }

     else return true;
}
```

```
function ValideDate()
 {
    if (isNaN(this.Jour) || isNaN(this.Mois) || isNaN(this.Annee))
     {
        alert ("Il faut une date de dépôt et une date de péremption VALIDES !");
        return false;
     }

    else
    {

    if (this.Jour<1 || this.Jour>31 || this.Mois<1 || this.Mois>12 || this.Annee < 1990 || this.Annee >
9999)
     {
       alert ("Erreur fondamentale de date!");
            return false;
     }
     else
     {
       if ((this.Mois==4 || this.Mois==6 || this.Mois==9 || this.Mois==11) && this.Jour==31)
       {
         alert ("Erreur de date : un mois de 30 jours en comporte 31 !");
         return false;
       }
       else
       {
         if (this.Mois==2 && this.Jour>29)
         {
           alert ("Erreur de format de date: le mois de février comporte plus de 28 ou 29 jours !");
           return false;
         }
         else
         {
           if (this.Mois==2 && this.Bisext()==false && this.Jour==29)
           {
           alert ("Erreur de format de date: le mois de février est non bisextile !");
           return false;
           }
           else
           {
             if (CompareDate(this, Maintenant) == true)
             {
              alert ("Erreur de date: une date (" + this.Jour+ '/' + this.Mois +'/' + this.Annee+ ") est
antérieure à celle d'aujourd'hui !");
              return false;
             }
             else
             {
              return true;
             }
           }
         }
       }
     }
    }
 }
```

```
function TTitre (Titre) {
    this.Titre = Titre;
    this.Valeur = Titre.value;
    this.ValideTitre = ValideTitre;
}

function ValideTitre()
 {
   if (this.Valeur == "")
       {
             window.alert ("Il faut un titre à cet avis");
        this.Titre.focus();
        return false;
       }
       else return true;
 }

function TContenu (Contenu)
 {
    this.Contenu = Contenu;
    this.Valeur = Contenu.value;
    this.ValideContenu = ValideContenu;
 }

function ValideContenu()
 {
    if (this.Valeur == "")
       {
       window.alert ("Il faut un contenu à cet avis");
       this.Contenu.focus();
       return false;
       }
       else return true;
 }

function TDestinataire (LM1, LM2, M3, Staff)
 {
    this.LM1 = LM1;
    this.LM2 = LM2;
    this.M3 = M3;
    this.Staff = Staff;
    this.ValideDestinataire = ValideDestinataire;
 }

function ValideDestinataire()
 {
   if (this.LM1.checked == false && this.LM2.checked == false && this.M3.checked == false &&
this.Staff.checked == false)
      {
       window.alert ("Il faut au moins un destinataire à cet avis");
       this.Staff.focus();
       return false;
      }
      else
      {
       return true;
      }
 }
```

```
function TAvis(Titre, Contenu, DateDebut, DateFin, Destinataire)
 {
    this.Titre = Titre;
    this.Contenu = Contenu;
    this.DateDebut = DateDebut;
    this.DateFin = DateFin;
    this.Destinataire = Destinataire;
    this.ValideTemps = ValideTemps;
    this.ValideAvis = ValideAvis;
 }

function ValideTemps()
 {
    if (CompareDate(this.DateDebut, this.DateFin) == false)
      {
         alert ("Erreur de date : la date de péremption de l'avis (" + this.DateFin.Jour +"/" +
this.DateFin.Mois + "/" + this.DateFin.Annee +                    ") doit être postérieure à celle de son dépôt
(" + this.DateDebut.Jour +"/"+ this.DateDebut.Mois +"/"+ this.DateDebut.Annee + ") !");
         return false;
      }
      else
      {
         return true;
      }
 }

function ValideAvis()
 {
   if (this.Titre.ValideTitre() == true && this.Contenu.ValideContenu() == true &&
this.DateDebut.ValideDate() == true
      && this.DateFin.ValideDate() == true && this.Destinataire.ValideDestinataire() == true &&
this.ValideTemps() == true)
      {
         return true;
      }
      else return false;
 }

function InitialDate(Form)
 {
     Form.DeJour.value= Maintenant.Jour;
     Form.DeMois.value= Maintenant.Mois;
     Form.DeAnnee.value= Maintenant.Annee;
     return true;
 }

function Initial(Document)
 {
     InitialDate(document.AvisForm);
 }

function Verif(Form)
  {
      LeTitre = new TTitre(Form.TitreAvis)
      LeContenu = new TContenu(Form.ContenuAvis)
      LaDateDebut = new TDate(Form.DeJour.value, Form.DeMois.value, Form.DeAnnee.value)
      LaDateFin = new TDate(Form.AuJour.value, Form.AuMois.value, Form.AuAnnee.value)
      LeDestinataire = new TDestinataire(Form.Ok1LM, Form.Ok2LM, Form.Ok3M, Form.OkStaff)
      LeAvis = new TAvis(LeTitre, LeContenu, LaDateDebut, LaDateFin, LeDestinataire)
```

```
    if (LeAvis.ValideAvis() == true)
      {

          Form.DeJour.value = LaDateDebut.Jour;
          Form.DeMois.value = LaDateDebut.Mois;
          Form.DeAnnee.value = LaDateDebut.Annee;

          Form.AuJour.value = LaDateFin.Jour;
          Form.AuMois.value = LaDateFin.Mois;
          Form.AuAnnee.value = LaDateFin.Annee;

          document.AvisForm.submit();
          alert ("Votre demande est envoyée au serveur !");
          return true;
     }
     else
     {
        return false;
     }
 }

function Annul(Form)
  {
    Form.TitreAvis.value ='';
    Form.ContenuAvis.value = '';

    Form.AuJour.value ='';
    Form.AuMois.value ='';
    Form.AuAnnee.value ='';

    Form.OkStaff.checked = false ;
    Form.Ok1LM.checked = false ;
    Form.Ok2LM.checked = false ;
    Form.Ok3M.checked = false ;

    InitialDate(Form);

    return true;
  }

//-->
</script>
</head>

<body bgcolor="#FFFFF0" onload="Initial(this.document)">

<h1 align="center">Ajout d'un nouvel avis aux valves</h1>

<hr color="#000000" noshade>

<form ACTION="cgi-bin/AjoutAvisValve.cgi" METHOD="POST" NAME="AvisForm">
  <p> </p>
  <table border="0" cellpadding="0">
    <tr>
      <td height="120" valign="top"><h2>Titre de l'avis :</h2>
      <p>             
                  
      <input TYPE="text" SIZE="85" NAME="TitreAvis" tabindex="1">
      </td>
    </tr>
```

```
    <tr>
      <td height="116" valign="top" align="left"><h2>Période de validité de l'avis :</h2>
      <p>           
      <big>à partir du :</big> 
      <input type="text" name="DeJour" size="02" tabindex="2"><big> / </big>
      <input type="text" name="DeMois" size="02" tabindex="3"><big> / </big>
      <input type="text" name="DeAnnee" size="4" tabindex="4">
                
      <big>jusqu'au :</big> 
      <input type="text" name="AuJour" size="02" tabindex="5"><big> / </big>
      <input type="text" name="AuMois" size="02" tabindex="6"><big> / </big>
       <input type="text" name="AuAnnee" size="4" tabindex="7"> </p>
      <p> </td>
    </tr>
    <tr>
       <td height="363" valign="top"><h2>Contenu de l'avis :</h2>
      <p>           
                   
         
      <textarea ROWS="14" COLS="86" NAME="ContenuAvis" tabindex="8"
       wrap=virtual></textarea></td>
    </tr>
    <tr>
       <td height="91" valign="top" align="left"><h2>Personnes concernées par l'avis :</h2>
      <p>           
                    
      <input TYPE="checkbox" VALUE="ON" NAME="OkStaff" tabindex="9">
      <big>le staff</big>
      <br><br>
                  
                  
      <input TYPE="checkbox" VALUE="ON" NAME="Ok1LM" tabindex="10">
      <big>les 1ères licences et maîtrises</big>
      <br><br>
                  
                  
       <input TYPE="checkbox" VALUE="ON" NAME="Ok2LM" tabindex="11">
       <big>les 2èmes licences et maîtrises</big>
       <br><br>
                   
                   

      <input TYPE="checkbox" VALUE="ON" NAME="Ok3M" tabindex="12">
      <big>les 3èmes maîtrises</big><br>
      </td>
    </tr>
    <tr>
      <td height="57" valign="middle"> <hr color="#000000" noshade size="1">
      <div align="center"><center><p><big>
       <input LANGUAGE="JavaScript" TYPE="button" VALUE="Afficher l'avis aux valves"
       NAME="ValiderBouton" tabindex="13" onClick="Verif(this.form);">
                   
                  
      <input TYPE="button" VALUE="Effacer le formulaire" NAME="EffacerBouton" tabindex="14"
      onClick="Annul(this.form);">
      </big></td>
    </tr>
  </table>
</form>
</body></html>
```

Annexe 4. Manuel d'installation des valves

Les valves électroniques se présentent sous la forme d'un seul fichier compressé. Voici la procédure nécessaire à leur installation et à leur mise en route.

1. Il faut d'abord copier le fichier et le décompresser. Deux types de fichier apparaissent : ceux dont l'extention est .cgi (les scripts) et ceux dont l'extension est .html (les pages Web). Un fichier « LisezMoi », qui reprend ce manuel d'installation, est également présent.

2. Les fichiers .cgi doivent être placés dans le répertoire « cgi-bin »

3. Les fichiers .html doivent être placés dans le répertoire « http-pub »

4. A partir du répertoire cgi-bin, il faut créer le répertoire « ../FichierValve »

5. Dans le répertoire cgi-bin, il faut que les scripts des valves soient attribués à l'utilisateur « nobody ». La commande « chown nobody *.cgi » doit donc être lancée : la configuration du système peut réserver cette commande au *root*.

6. Le répertoire /FichierValve doit lui aussi être attribué à « nobody ». Il suffit alors de permettre les droits d'écriture et de lecture pour son seul propriétaire.

7. Il faut, enfin, demander au *Cron Daemon* d'exécuter tous les jours, à l'heure la plus calme possible sur le réseau, le script MiseAJourValve.cgi. Nous avions opté, durant les tests, pour 1h10 du matin.

8. Pour rendre les valves opérationnelles, il suffit de lancer le script MiseAJourValve.cgi, par simple ligne de commande. Automatiquement, les fichiers des valves seront créés dans le répertoire /FichierValve. Le script, puisqu'il est confronté à des fichiers inexistants, enverra un *mail* par fichier au Webmaster pour lui signaler leur création : ce dernier recevra donc huit *mails*.

9. Par défaut, le Webmaster des valves est désigné comme « www@info.fundp.ac.be ». Nous avons abordé, dans notre mémoire, comment modifier le Webmaster : il suffit de changer, dans chaque script, la valeur d'une constante (Destinataire). Le sujet des mails envoyé sera, par défaut, « ValveElectronique/ControleAutomatique » : pour l'adapter, il suffit de modifier la valeur de la constante Sujet. De même, le nom du répertoire /FichierValve peut être modifié, lui aussi : c'est la valeur de la constante CheminFichier qui doit être changée dans chaque script.

Pour des modifications plus importantes, nous renvoyons au texte de notre mémoire.